IBM

Series/1

GA34-0289-1

File No. S1-16

IBM Series/1
4956 Processor Models E and E10
Description

GA34-0289-1

File No. S1-16

IBM Series/1
4956 Processor Models E and E10
Description

Second Edition (January 1986)

Use this publication for the purpose stated in the preface.

Changes are periodically made to the information herein; any such changes will be reported in subsequent revisions or Technical Newsletters.

It is possible that this material may contain reference to, or information about, IBM products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such IBM products, programming, or service in your country.

Publications are not stocked at the address given below. Request for copies of IBM publications should be made to your IBM represenative or the IBM branch office serving your locality.

This publication could contain technical inaccuracies or typographical errors. A form for readers' comments is provided at the back of this publication. If the form has been removed, address your comments to IBM Corporation, Information Development, Department 28E, Internal Zip 1803, P. O. Box 1328, Boca Raton, Florida 33429-1328. IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Preface

This publication describes the unique functional characteristics and the optional features of the IBM Series/1 4956 Processor Models E and E10. This publication also provides reference information about the possible configurations and feature operations of the processor. Refer to the *IBM Series/1 Principles of Operation*, GA34-0152, for the common Series/1 processor functional characteristics and instructions.

This publication is intended primarily as a reference manual for experienced programmers who require machine code information to plan, correct, and modify programs written in the assembler language. The reader should understand data processing terminology and be familiar with binary and hexadecimal numbering systems.

Chapter 1, "Introduction," contains a general description of the processor, processor storage, and processor features. This chapter also contains information about changes to the programming instruction set.

Chapter 2, "Processor Storage Addressing Using the Relocation Translator," describes the relocation translator, including:

- Relocation addressing
- Storage-protection mechanism
- Error-recovery considerations.

Chapter 3, "Console," describes the keys, switches, and indicators for the basic console and the optional programmer console. Typical manual operations, such as storing into and displaying processor storage, are presented.

Chapter 4, "Diagnose (DIAG) Instruction," describes the Diagnose instruction.

Appendix A, "Instruction Execution Times," contains information for determining instruction execution times and instruction throughput.

Appendix B, "Software Notes," lists some software notes for the processor.

Appendix C, "Error Log," describes the error log and explains its use as an aid in isolating errors.

Prerequisite Publication

For a description of the processor architecture and a detailed description of the instruction set for the IBM Series/1 processors, refer to the *IBM Series/1 Principles of Operation*, GA34-0152.

Contents

Chapter 1. Introduction

The IBM Series/1 4956 Processor Models E and E10 are compact general-purpose computers. The models are the same, except for basic storage. Model E has 256 or 512 kilobytes of basic storage; Model E10 has 1024 kilobytes of basic storage.

The processor is microcode-controlled for both automatic functions and program instruction functions. It occupies the full width of a standard 483-millimeter (19-inch) rack (see Figure 1-1). It contains thirteen card sockets for data channel features and a channel repower card. Three of the thirteen card sockets can also be used for additional processor storage cards.

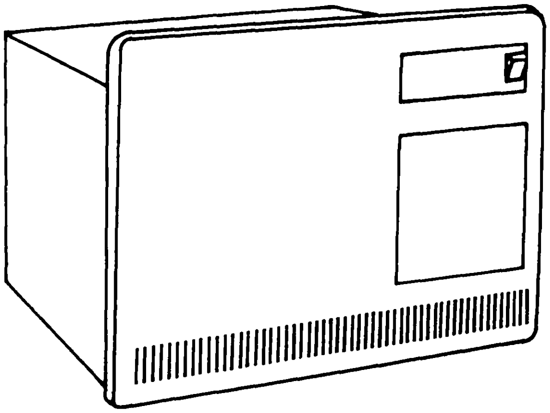


Figure 1-1. IBM Series/1 4956 Processor Models E and E10

The processor has the following characteristics:

- Four priority interrupt levels, with independent registers and status indicators for each level.
- Automatic and program-controlled level switching.
- An instruction set that includes stacking and linking facilities, multiply and divide, variable-field-length byte operations, and a variety of arithmetic and branching instructions.
- Supervisor and problem states.
- A basic console that is a standard feature; a programmer console that is an optional feature.
- Basic main storage and up to three additional storage increments of either 256 kilobytes, 512 kilobytes, or 1024 kilobytes, with a maximum possible storage of 2048 kilobytes (2 megabytes).
- A storage address relocation translator that allows addressing of main storage larger than 64 kilobytes.
- Program-controlled mode switching that allows the user to specify the maximum mappable storage range:
  - 512K bytes in 3-bit mode
  - 1024K bytes (1 MB) in 4-bit mode
- Prefetching instruction stream. (Refer to Appendix B.)
- An error correction code (ECC) that is implemented on the storage card to provide the capability for single-bit error correction and double-bit error detection.
- An error log that provides a history of errors that have occurred since power-on. (Refer to Appendix C.)
- A clock/comparator with four instructions provided to set or copy the clock and comparator.
- Channel capability as follows:
  - Asynchronous, multidropped channel
  - 256 input/output (I/O) devices can be addressed
  - Direct program control and cycle-steal operations
  - Maximum burst output data rate of 1.11 million 16-bit words per second (see Note)
  - Maximum burst input data rate of 1.54 million 16-bit words per second (see Note).

Note: The burst output and burst input data rates are reduced from the values shown by data channel attachment characteristics, channel loading during instruction processing, channel repowering, and processor storage refresh requirements.

- Throughput that is approximately 50% greater than previous models of the 4956.

- A redefined 16-bit address key register (AKR).

- A redefined 16-bit processor status word (PSW) register.

- Several instructions modified for 4-bit mode:

  - Set Address Key Register (SEAKR)
  - Copy Address Key Register (CPAKR)
  - Set Segmentation Registers (SESR)
  - Copy Segmentation Registers (CPSR)
  - Enable (EN)
  - Disable (DIS)
  - Copy Processor Status and Reset (CPPSR)

  Note: For an explanation of the standard Series/1 instruction set, refer to *IBM Series/1 Principles of Operation,* GA34-0152.

- Three new instructions in the instruction set:

  - Address Resolution with Indirect Branch (ARIB)
  - Address Resolution with Indirect Branch—On (ARIBON)
  - Address Resolution with Indirect Branch—Off (ARIBOFF)

Processor Description

The basic processor includes the processor card, a basic storage card, and a basic console. Figure 1-2 shows a block diagram of the processor and an IBM Series/1 4959 Input/Output Expansion Unit.

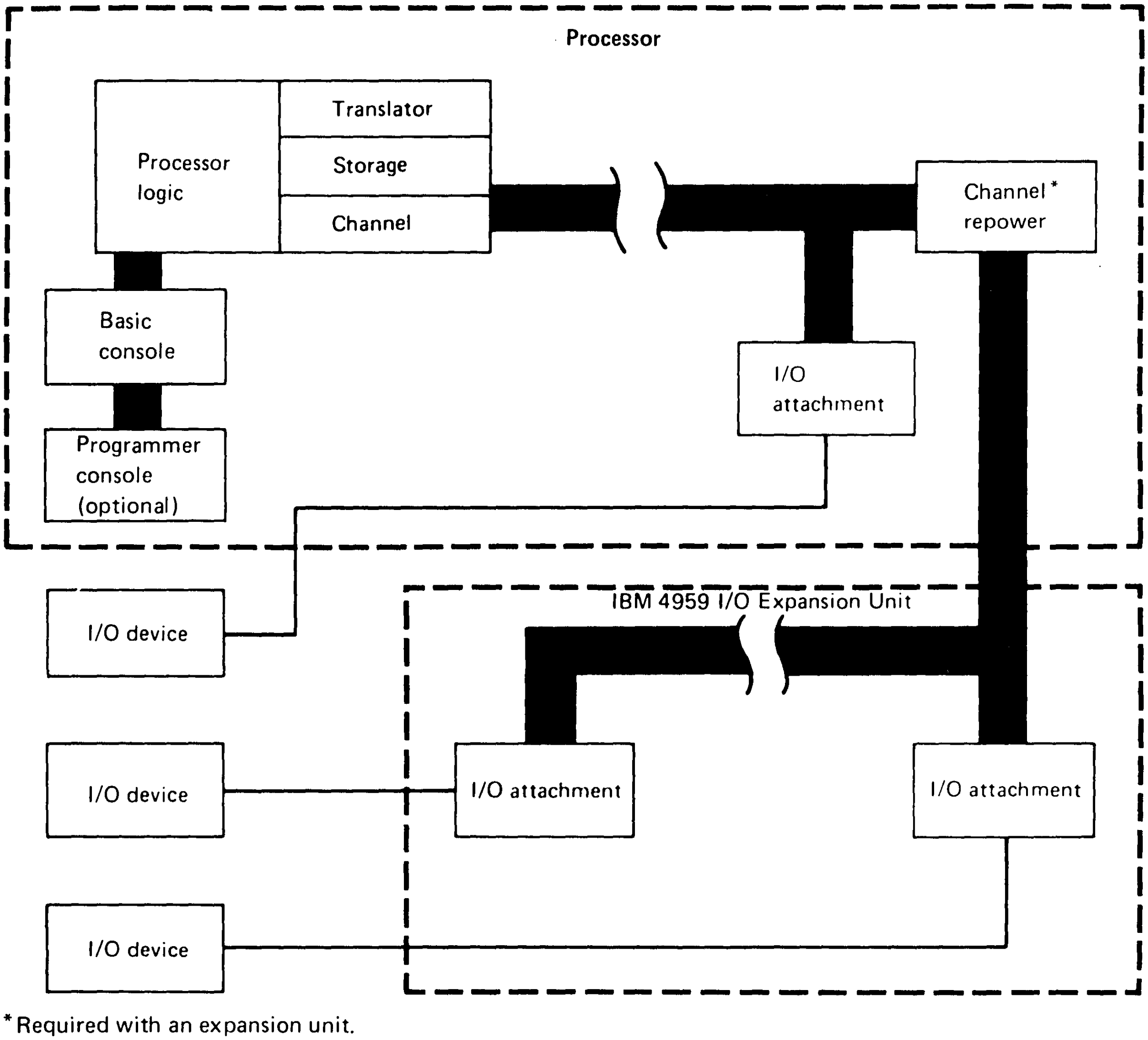


Figure 1-2. Block Diagram of the Processor and an IBM 4959 I/O Expansion Unit

Four priority interrupt levels (0–3) are implemented in the processor. Each level has an independent set of machine registers. Level switching can occur in two ways: (1) by program control, or (2) automatically upon acceptance of an I/O interrupt request. The interrupt mechanism provides 256 unique entry points for I/O devices.

> **Note:** A Prepare command to levels 4–15 is executed so that condition code reporting occurs; however, the Prepare command is not executed at the addressed device and effectively results in a no-operation.

The processor instruction set contains a variety of instruction types. These include:

- bit manipulation
- shift
- branch
- register immediate
- storage immediate
- register to register
- register to storage
- storage to register
- storage to storage
- system register to storage
- multiple register to storage
- variable byte field.

Supervisor and problem states are implemented, with appropriate privileged instructions for the supervisor.

The basic console is intended for dedicated systems that are used in a primarily unattended environment. Only minimal controls are provided. A programmer console, which can be added as a feature, provides a variety of indicators and controls for operator-oriented systems.

An error correction code (ECC) is implemented on the storage card. ECC gives the storage card the capability of single-bit error correction and double-bit error detection. ECC provides the user a higher system availability.

> **Note:** When a double-bit error in storage is detected during a processor read, a machine check interrupt occurs with PSW bit 8 set to 1 (storage parity error).

There is no storage-protect feature in the processor. However, there is a read-only protect capability provided by the address translator when it is enabled.

> **Note:** Execution of the Set Storage Key (SESK) and Copy Storage Key (CPSK) instructions results in a no-operation.

I/O devices are attached to the processor through the processor data channel. The data channel directs the flow of information between the I/O devices, the processor, and main storage. The data channel supports a maximum of 256 addressable devices.

The data channel supports:

- **Direct program control operations**. Each Operate I/O instruction transfers a byte or word of data between main storage and the device. The operation may or may not terminate in an interrupt.

- **Cycle-steal operations**. Each Operate I/O instruction initiates multiple data transfers between main storage and the device. The maximum cycle-steal transfer per device control block (DCB) is 65,535 bytes. Cycle-steal operations are overlapped with processor operations and always terminate in an interrupt.

- **Interrupt servicing**. Interrupt requests from the devices, along with cycle-steal requests, are presented and polled concurrently with data transfers.

Storage Cards

The processor supports three different size storage cards: 256 KB, 512 KB, and 1024 KB. The processor has a maximum of 2048 KB of storage. Any combination of storage cards may be used (with the exception of one 512 KB card with three 256 KB cards) to obtain the desired system storage size, up to the maximum of 2048 KB. (The relocation translator must be enabled to select addresses above 64K bytes.)

Several other rules apply for proper storage card plugging:

- All storage cards are plugged in decrementing order from the processor card.
- No card sockets may be left unused between storage cards.
- Storage cards must be plugged in the order for which they are jumpered.
- Any card socket that is not used for additional storage may be used for any data channel feature or a repower card.

Card Plugging Assignments

The processor unit contains power and space for additional features. The IBM Series/1 4959 Input/Output Expansion Unit and the IBM Series/1 4965 Storage and I/O Expansion Unit are available for adding additional features, if desired. Figure 1-3 shows the card plugging assignments for the processor.

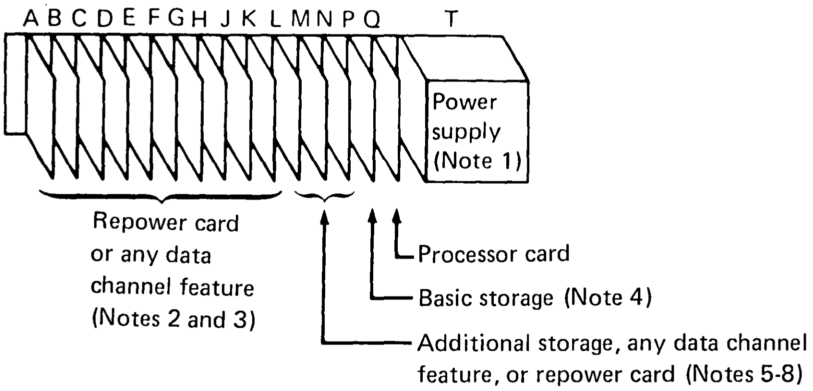

A B C D E F G H J K L M N P Q
T
Power supply (Note 1)
Repower card or any data channel feature (Notes 2 and 3)
Processor card
Basic storage (Note 4)
Additional storage, any data channel feature, or repower card (Notes 5-8)

Figure 1-3. Card Plugging Assignments

Notes:

1. The pluggable high-frequency power supply plugs into card socket T.

2. If a channel repower card is used, it must be plugged to the left of and adjacent to the leftmost I/O card installed.

3. A maximum of five serially-connected channel repower features can be driven by each processor. Any processor system that includes an IBM I/O expansion unit with the two-channel switch feature is limited to three channel repower features.

4. The processor contains 256 KB (kilobytes), 512 KB, or 1024 KB of basic storage in socket P.

5. The processor supports three different size storage cards; 256 KB, 512 KB, and 1024 KB. Sockets N, M, and L are available for additional storage cards. Any combination of storage cards is permitted (except the combination of a single 512KB card with three 256KB cards), up to the maximum storage capacity of 2048 KB. Any card socket that is not used for additional storage may be used for any data channel feature or a repower card.

6. All storage cards are plugged in decrementing order from the processor card.

7. No card sockets may be left unused between storage cards.

8. Storage cards must be plugged in the order for which they are jumpered.

Input/Output Units, I/O Features, and Processor Options

A variety of I/O units and features, plus several processor options, are available for use with the processor. For a list and description of system units and features, refer to the *IBM Series/1 System Selection Guide*, GA34-0143, and the *IBM Series/1 Digest*, G360-0061. Detailed information about I/O units and features can be found in separate publications. The order numbers for these publications are contained in the *IBM Series/1 Graphic Bibliography*, GA34-0055.

The floating-point feature is one of the available options. If the floating-point feature is installed, refer to Appendix A for instruction execution times. For a detailed description of this feature, refer to the *IBM Series/1 Principles of Operation*, GA34-0152.

Program-Controlled Mode Switching

Program-controlled mode switching allows the user to specify the maximum mappable storage range:

- In 3-bit mode, the maximum mappable storage is 512K bytes. Operand 1 key (OP1K), operand 2 key (OP2K), and instruction space key (ISK) are specified in three bits. The translator uses 3 bits from the active address key and the five high-order bits of the logical address to select one of 256 segmentation registers when the translator is enabled.

- In 4-bit mode the maximum mappable storage is 1024K bytes (1 MB). In 4-bit mode, OP1K, OP2K, and ISK are specified in four bits. Eight additional stacks of 32 registers per stack are available for I/O operations. This allows 512K bytes of storage to be mapped for I/O only, independent of program mapping.

Bit 7 of the processor status word (PSW) indicates which mode the processor is currently in:

- When bit 7 is a 0, the processor is in 3-bit mode.
- When bit 7 is a 1, the processor is in 4-bit mode.

When both bit 7 and bit 14 (translator bit) are 1's, the I/O translator is enabled.

Address Key Register (AKR)

The address key register (AKR) contains 16 bits. It is formatted as follows:

For 3-bit mode:

Bit	**Data**
0	Equate operand spaces
1	0
2	0
3	0
4	0
5	OP1K bit 0
6	OP1K bit 1
7	OP1K bit 2
8	0
9	OP2K bit 0
10	OP2K bit 1
11	OP2K bit 2
12	0
13	ISK bit 0
14	ISK bit 1
15	ISK bit 2

For 4-bit mode:

Bit	**Data**
0	Equate operand spaces
1	0
2	0
3	0
4	OP1K bit 0
5	OP1K bit 1
6	OP1K bit 2
7	OP1K bit 3
8	OP2K bit 0
9	OP2K bit 1
10	OP2K bit 2
11	OP2K bit 3
12	ISK bit 0
13	ISK bit 1
14	ISK bit 2
15	ISK bit 3

Processor Status Word (PSW) Register

The processor status word (PSW) register contains 16 bits. It is formatted as follows:

Bit	Data
0	Specification check
1	Invalid storage address
2	Privilege violation
3	Protect check
4	Invalid function
5	Floating point exception
6	Stack exception
7	4-bit mode enabled
8	Storage parity check
9	0
10	Processor control check
11	I/O check
12	Sequence indicator
13	Auto IPL
14	Translator enabled
15	Power thermal warning

Modified Instructions

Modifications have been incorporated into several Series/1 instructions pertaining to their use with the processor. These modifications apply only for 4-bit mode operation; they do not apply for 3-bit mode operation. The instructions that have been modified are:

- Set Address Key Register (SEAKR)
- Copy Address Key Register (CPAKR)
- Set Segmentation Registers (SESR)
- Copy Segmentation Registers (CPSR)
- Enable (EN)
- Disable (DIS)
- Copy Processor Status and Reset (CPPSR).

Note: For an explanation of the standard Series/1 instruction set, refer to *IBM Series/1 Principles of Operation* GA34-0152.

Set Address Key Register (SEAKR)

The Set Address Key Register (SEAKR) instruction has two formats:

- System register/register format for operations that load data from a specified register into the AKR.
- System register/storage format for operations that load data from main storage into the AKR.

System Register/Register Format

The system register/register format is:

Mnemonic	**Syntax**	**Instruction name**	**K-field**
SEAKR	reg	Set Address Key Register	011

Extended mnemonic	**Syntax**	**Instruction name**	**K-field**
SEISK	reg	Set Instruction Space Key	000
SEOOK	reg	Set Operand 1 Key	010
SEOTK	reg	Set Operand 2 Key	001

Op code	*K*	*R*	*Function*
0 1 1 1 1			1 0 0 1 0
0 4	*5 7*	*8 10*	*11 15*

The address key register (AKR) field specified by the K-field is loaded from the register specified by the R-field. The contents of the register are not changed.

Note: The K-field can specify either a field within the AKR or an entire AKR.

For 3-bit mode:

K-field	Address key register field name	Bits
000	Instruction space key	13–15
001	Operand 2 key	9–11
010	Operand 1 key	5–7
011	Address key register	0–15
100	Should not be used	
101	Should not be used	
110	Should not be used	
111	Should not be used	

For 4-bit mode:

K-field	Address key register field name	Bits
000	Instruction space key	12–15
001	Operand 2 key	8–11
010	Operand 1 key	4–7
011	Address key register	0–15
100	Should not be used	
101	Should not be used	
110	Should not be used	
111	Should not be used	

If the K-field specifies a specific field within the AKR, bits 13–15, for 3-bit mode, or bits 12–15, for 4-bit mode, from the register specified by the R-field are loaded into the AKR field. If the K-field specifies the entire AKR, bits 0–15 from the specified register are loaded into the AKR.

Indicators: The indicators are not changed.

Program Checks: This instruction format has the following program check:

- **Privilege violate** — The instruction is encountered while in problem state. The instruction is suppressed and a program check interrupt occurs with privilege violate set in the PSW.

System Register/Storage Format

Mnemonic	Syntax	Instruction name	K-field
SEAKR	addr4	Set Address Key Register	011

Extended mnemonic	Syntax	Instruction name	K-field
SEISK	addr4	Set Instruction Space Key	000
SEOOK	addr4	Set Operand 1 Key	010
SEOTK	addr4	Set Operand 2 Key	001

Op code	K	RB	AM	Function
0 1 0 1 1				0 0 1 0
0 4	5 7	8 9	10 11	12 15

Address/Displacement	
Displacement 1	Displacement 2
16 23	24 31

The address key register (AKR) field specified by the K-field is loaded from the word location in main storage that is specified by the effective address. The contents of the word in main storage are not changed.

> **Note:** The K-field can specify either a field within the AKR or an entire AKR.

For 3-bit mode:

K-field	Address key register field name	Bits
000	Instruction space key	13–15
001	Operand 2 key	9–11
010	Operand 1 key	5–7
011	Address key register	0–15
100	Should not be used	
101	Should not be used	
110	Should not be used	
111	Should not be used	

For 4-bit mode:

K-field	Address key register field name	Bits
000	Instruction space key	12–15
001	Operand 2 key	8–11
010	Operand 1 key	4–7
011	Address key register	0–15
100	Should not be used	
101	Should not be used	
110	Should not be used	
111	Should not be used	

If the K-field specifies a specific field within the AKR, bits 13–15, for 3-bit mode, or bits 12–15, for 4-bit mode, from the word location in main storage are loaded into the AKR field. If the K-field specifies the entire AKR, bits 0–15 from the word location in main storage are loaded into the AKR.

Indicators: The indicators are not changed.

Program Checks: This instruction format has the following program checks:

- **Invalid storage address** — One or more words of the instruction or the effective address are outside the installed storage size of the system. The instruction ends and a program check interrupt occurs with invalid storage address set in the PSW.

- **Privilege violate** — The instruction is encountered while in problem state. The instruction is suppressed and a program check interrupt occurs with privilege violate set in the PSW.

- **Specification check** — The effective address or indirect address results in an even-byte boundary violation. The instruction ends and a program check interrupt occurs with specification check set in the PSW.

Copy Address Key Register (CPAKR)

The Copy Address Key Register (CPAKR) instruction has two formats:

- System register/register format for operations that load data from the AKR into a specified register.

- System register/storage format for operations that load data from the AKR into main storage.

System Register/Register Format

The system register/register format is:

Mnemonic	**Syntax**	**Instruction name**	**K-field**
CPAKR	reg	Copy Address Key Register	011

Extended mnemonic	**Syntax**	**Instruction name**	**K-field**
CPISK	reg	Copy Instruction Space Key	000
CPOOK	reg	Copy Operand 1 Key	010
CPOTK	reg	Copy Operand 2 Key	001

Op code	*K*	*R*	*Function*
0 1 1 1 1			1 1 0 1 0
0 4	*5 7*	*8 10*	*11 15*

The contents of the address key register (AKR) field specified by the K-field are loaded into the register specified by the R-field. The contents of the AKR are not changed.

Note: The K-field can specify a field within the AKR or the entire AKR.

For 3-bit mode:

K-field	**Address key register field name**	**Bits**
000	Instruction space key	13–15
001	Operand 2 key	9–11
010	Operand 1 key	5–7
011	Address key register	0–15
100	Should not be used	
101	Should not be used	
110	Should not be used	
111	Should not be used	

For 4-bit mode:

K-field	Address key register field name	Bits
000	Instruction space key	12–15
001	Operand 2 key	8–11
010	Operand 1 key	4–7
011	Address key register	0–15
100	Should not be used	
101	Should not be used	
110	Should not be used	
111	Should not be used	

If the K-field specifies a specific field within the AKR, the specified field is loaded into bits 13–15, for 3-bit mode, or bits 12–15, for 4-bit mode, of the register specified in the R-field. Bits 0–12, for 3-bit mode, or bits 0–11, for 4-bit mode, are set to 0's. If the K-field specifies the entire AKR, the AKR is loaded into the register.

Indicators: The indicators are not changed.

Program Checks: This instruction format has the following program check:

- **Privilege violate** — The instruction is encountered while in problem state. The instruction is suppressed and a program check interrupt occurs with privilege violate set in the PSW.

System Register/Storage Format

The system register/storage format is:

Mnemonic	Syntax	Instruction name	K-field
CPAKR	addr4	Copy Address Key Register	011

Extended mnemonic	Syntax	Instruction name	K-field
CPISK	addr4	Copy Instruction Space Key	000
CPOOK	addr4	Copy Operand 1 Key	010
CPOTK	addr4	Copy Operand 2 Key	001

Op code	*K*	*RB*	*AM*	*Function*
0 1 0 1 1				1 0 1 0
0 4	*5 7*	*8 9*	*10 11*	*12 15*

Address/Displacement	
Displacement 1	Displacement 2
16 23	*24 31*

The contents of the address key register (AKR) field specified by the K-field are stored in the word location specified by the effective address. The contents of the AKR are not changed.

> **Note:** The K-field can specify a field within the AKR or the entire AKR.

For 3-bit mode:

K-field	Address key register field name	Bits
000	Instruction space key	13–15
001	Operand 2 key	9–11
010	Operand 1 key	5–7
011	Address key register	0–15
100	Should not be used	
101	Should not be used	
110	Should not be used	
111	Should not be used	

For 4-bit mode:

K-field	Address key register field name	Bits
000	Instruction space key	12–15
001	Operand 2 key	8–11
010	Operand 1 key	4–7
011	Address key register	0–15
100	Should not be used	
101	Should not be used	
110	Should not be used	
111	Should not be used	

If the K-field specifies a specific field within the AKR, the specified field is stored in bits 13–15, for 3-bit mode, or bits 12–15, for 4-bit mode, of the word location in main storage. Bits 0–12, for 3-bit mode, or bits 0–11, for 4-bit mode, of the word in main storage are set to 0's. If the K-field specifies the entire AKR, the AKR is stored in the word location in main storage.

Indicators: The indicators are not changed.

Program Checks: This instruction format has the following program checks:

- **Invalid storage address** — One or more words of the instruction or the effective address are outside the installed storage size of the system. The instruction ends and a program check interrupt occurs with invalid storage address set in the PSW.

- **Privilege violate** — The instruction is encountered while in problem state. The instruction is suppressed and a program check interrupt occurs with privilege violate set in the PSW.

- **Specification check** — The effective address or indirect address results in an even-byte boundary violation. The instruction ends and a program check interrupt occurs with specification check set in the PSW.

Set Segmentation Register (SESR)

The syntax for this instruction is:

SESR   reg,addr4

Op code	R	RB	AM	Function
0 1 0 1 1				0 0 0 1
0 4	*5 7*	*8 9*	*10 11*	*12 15*

Address/Displacement	
Displacement 1	Displacement 2
16 23	*24 31*

This instruction loads the contents of one or more doubleword storage locations, the first of which is specified by the effective address, into the segmentation registers specified by the contents of the register specified by the R-field.

For processors with 3-bit keys enabled, the format of the register specified by the R-field is:

0 4	*5 7*	*8 9*	*10 14*	*15*
Logical sector	*Key*	0 0	*Count*	0
Key bits 0 1 2				

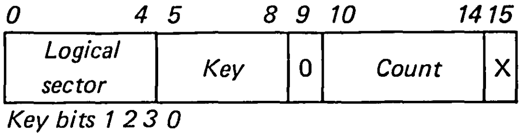
Bits 0–7 form the number of the segmentation register to be loaded (0–255). This number is comprised of 3-bits from the address key (values 0–7) and the five high-order bits of the logical storage address which is the logical segment (values 0–31). Bits 8, 9, and 15 of the register are reserved and must be set to 0's.

Bits 10–14 specify the number of contiguous segmentation registers to be set from contiguous doubleword storage locations.

Notes:

1. The count is equal to count plus 1. For example a count of 0 loads one register.
2. If AM=01, the register selected by the RB field is incremented by 4 for each segmentation register set.

For 4-bit mode, the format of the register specified by the R-field is:



> **Note:** Bit 8 is the most significant bit in the key field, and bit 7 is the least significant bit. For example, 0011 in the key field would indicate stack 9.

Bits 0–8 form the number of the segmentation registers to be referenced (0–511) where the value of key bit 0 provides a select control between the first and second 256 segment register groups. This number is comprised of four bits from the address key (values 0–15) and the five high-order bits of the logical storage address, which is the logical segment (values 0–31). Bit 9 of the register is reserved and must be set to 0.

Bits 10–14 specify the number of contiguous segmentation registers to be set from contiguous doubleword locations (count plus 1). The instruction ends when the count is exhausted or when register 31 is loaded.

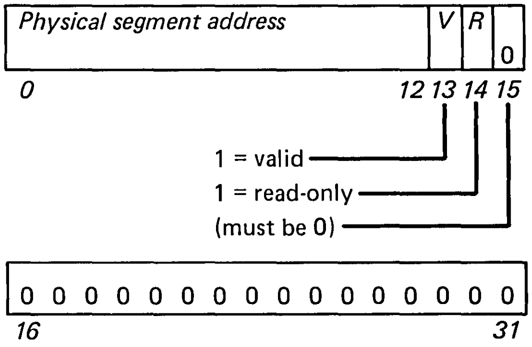
Bit 15, when a 1, indicates that the segmentation register stack referenced is the one reserved for I/O. Bit 8 (in the key field) is ignored in this case.

Notes:

1. If AM=01, the register selected by the RB field is incremented by 4 for each segmentation register set.

2. All indicators are unchanged.

The first word (bits 0–15) of the specified doubleword that is loaded into the selected segmentation register has the following format:



The segment address (bits 0–12) contains the high-order bits of the physical address.

Bit 13, if a 1, signifies that the contents of the segmentation register are valid, and translation can be performed. If an attempt is made to use a segmentation register with bit 13 set to 0, a program check interrupt occurs with invalid storage address set in the PSW.

Bit 14, if a 1, signifies that the block is read-only. If an attempt is made to write into the block when bit 14 of the associated segmentation register is a 1 and while in problem state, a program check interrupt occurs, with protect check set in the PSW. When in supervisor state or on a cycle-steal access, bit 14 is ignored. The contents of main storage can be changed.

The second word (bits 16–31) of the specified doubleword should be set to 0's.

Indicators: The indicators are not changed.

Program Checks: This instruction has the following program checks:

- **Invalid function** — In the supervisor state, an attempt has been made to invoke this instruction when the translator is not enabled.

- **Invalid storage address** — One or more words of the instruction or the effective address are outside the installed storage size of the system. The instruction ends and a program check interrupt occurs with invalid storage address set in the PSW.

- **Privilege violate** — The instruction is encountered while in problem state. The instruction is suppressed and a program check interrupt occurs with privilege violate set in the PSW.

- **Specification Check** — The effective address or indirect address results in an even-byte boundary violation. The instruction ends and a program check interrupt occurs with specification check set in the PSW.

Copy Segmentation Register (CPSR)

The syntax for the register/storage format is:

CPSR reg,addr4

Op code	R	RB	AM	Function
0 1 0 1 1				1 0 0 1
0 4	5 7	8 9	10 11	12 15

Address/Displacement	
Displacement 1	Displacement 2
16 23	24 31

This instruction stores the contents of one or more segmentation registers, specified by the contents of the register specified by the R-field, into contiguous doubleword storage locations, the first of which is specified by the effective address.

In the 3-bit mode, the format of the general register specified by the R-field is:

0 4	5 7	8 9	10 14	15
Logical sector	Key	0 0	Count	0
	Key bits 0 1 2			

Bits 0–7 form the number of the segmentation register to be copied (0–255). This number is comprised of 3-bits from the address key (values 0–7) and the five high-order bits of the logical storage address which is the logical segment (values 0–31). Bits 8, 9 and 15 of the register are reserved and must be set to 0.

Bits 10–14 specify the number of contiguous segmentation registers to be copied to contiguous doubleword storage locations and is equal to the count plus 1 (for example a count of zero loads one doubleword storage location). The instruction ends when the count is exhausted or when register 31 of the key space has been copied.

> **Note:** If AM=01, the register selected by the RB field is incremented by 4 for each segmentation register copied.

For processors with 4-bit mode enabled, the format of the register specified by the R-field is:

0 4	5 8	9	10 14	15
Logical sector	Key	0	Count	X
	Key bits 1 2 3 0			

> **Note:** Bit 8 is the most significant bit in the key field, and bit 7 is the least significant bit. For example, 0011 in the key field would indicate stack 9.

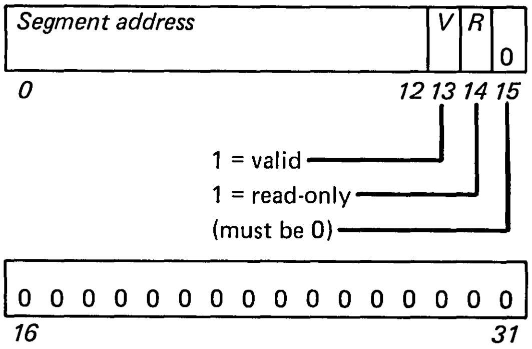
Bits 0–8 form the number of the segmentation register to be referenced (0–511) where the value of key bit 0 provides a select control between the first and second 256 segment register groups. This number is comprised of 4-bits from the address key (values 0–15) and the five high-order bits of the logical storage address which is the logical segment (values 0–31). Bit 9 of the register is reserved and must be set to 0.

Bit 15, when set to 1, indicates that the segmentation register stack referenced is the one reserved for I/O. Bit 8 (in the key field) is ignored in this case.

Bits 10–14 specify the number of contiguous segmentation registers to be copied to contiguous doubleword storage locations, with the count equal to the count plus 1 (for example if the count is zero, one doubleword storage location is loaded).

> **Note:** If AM=01, the register selected by the RB field is incremented by 4 for each segmentation register copied.

The first word of the specified doubleword that is copied from the selected segmentation register has the following format:



The segment address (bits 0–12) contains the high-order bits of the physical address, which is used by the translator to select a 2K-byte block of main storage.

Bit 13, if a 1, signifies that the contents of the segmentation register is valid, and the translation can be performed. If an attempt is made to use a segmentation register in which bit 13 is a 0, a program check interrupt occurs,with invalid storage address set in the PSW.

Bit 14, if a 1, signifies that the block is read-only. If an attempt is made to write into the block when bit 14 of the associated segmentation register is a 1 and while in problem state, a program check interrupt occurs, with protect check set in the PSW. When in supervisor state or on a cycle-steal access, bit 14 is ignored; the contents of main storage can be changed.

The second word (bits 16–31) of the specified doubleword should be set to 0's.

Indicators: The indicators are not changed.

Program Checks: This instruction has the following program checks:

- **Invalid function** — In the supervisor state, an attempt has been made to invoke this instruction when the translator is not enabled.
- **Invalid storage address** — One or more words of the instruction or the effective address are outside the installed storage size of the system. The instruction ends and a program check interrupt occurs with invalid storage address set in the PSW.
- **Privilege violate** — The instruction is encountered while in problem state. The instruction is suppressed and a program check interrupt occurs with privilege violate set in the PSW.
- **Specification check** — The effective address or indirect address results in an even-byte boundary violation. The instruction ends and a program check interrupt occurs with specification check set in the PSW.

Copy Processor Status and Reset (CPPSR)

The syntax for this instruction is:

CPPSR addr4

Op code		RB	AM	Function
0 1 0 1 1	0 0 0			1 1 1 1
0 4	5 7	8 9	10 11	12 15

Address/Displacement	
Displacement 1	Displacement 2
16 23	24 31

The contents of the processor status word (PSW) are stored at the word location in main storage specified by the effective address.

This instruction resets PSW bits 0–6 and 8–12. of Bits 7 and 13–15 are not changed. Bits 5–7 of the instruction are not used and should be set to 0's.

Indicators: The indicators are not changed.

Program Checks: This instruction has the following program checks:

- **Invalid storage address** — One or more words of the instruction or the effective address are outside the installed storage size of the system. The instruction ends and a program check interrupt occurs with invalid storage address set in the PSW.

- **Privilege violate** — The instruction is encountered while in problem state. The instruction is suppressed and a program check interrupt occurs with privilege violate set in the PSW.

- **Specification check** — The effective address or indirect address results in an even-byte boundary violation. The instruction ends and a program check interrupt occurs with specification check set in the PSW.

Enable (EN)

The syntax for this instruction is:

EN ubyte

Op code	Func	Parameter
0 1 1 0 0	0 1 0	
0 4	5 7	8 15

The parameter field bits have the following significance:

Bit	Significance
8	Not used
9	Not used
10	Enable 4-bit mode (PSW bit 7=1)
11	Not used
12	Enable storage protect
13	Enable equate operand spaces (AKR bit 0 set to 1)
14	Enable translator (PSW bit 14 set to 1)
15	Enable summary mask (LSR bit 11 set to 1)

Note: Bits not used must be set to 0's.

If bit 12 is set to 1, the relocation translator (if enabled) is disabled and bit 14 is not checked.

If bit 14 is set to 1, and bit 12 is set to 0, the relocation translator is enabled.

If parameter bit 14 and parameter bit 10 are both 1's or if parameter bit 14 and PSW bit 7 are both 1's, the I/O translator is enabled.

If parameter bit 14 is a 1, parameter bit 10 is a 0, and PSW bit 7 is a 0, the I/O translator is not enabled.

Indicators: The indicators are not changed.

Program Checks: This instruction has the following program check:

- **Privilege violate** — The instruction is encountered while in problem state. The instruction is suppressed and a program check interrupt occurs with privilege violate set in the PSW.

Disable (DIS)

The syntax for this instruction is:

DIS ubyte

Op code	*Func*	*Parameter*
0 1 1 0 0	0 1 1	
0 4	*5 7*	*8 15*

The bits in the parameter field have the following significance:

Bit	**Significance**
8	Not used
9	Not used
10	Disable 4-bit mode (PSW bit 7=0)
11	Not used
12	Disable storage protect
13	Disable equate operand spaces (AKR bit 0 set to 0)
14	Disable translator (PSW bit 14 set to 0)
15	Disable summary mask (LSR bit 11 set to 0)

Note: Bits not used must be set to 0's.

If a Disable instruction immediately follows an Enable summary mask instruction, the interrupt disable function may occur prior to the time that an interrupt can be accepted. Thus, at least one other instruction (for example, no-op) must be inserted between the Enable summary mask and Disable instructions to ensure the occurrence of the interrupt.

If parameter bit 14 is set to 1 and the relocation translator is enabled (bit 14 of the PSW is a 1), then the translator is disabled and bit 14 of the PSW is set to a 0. If parameter bit 14 is a 1, the I/O translator is always disabled.

If PSW bit 14 is a 1 and parameter bit 10 is a 1, the I/O translator is disabled.

Indicators: The indicators are not changed.

Program Checks: This instruction has the following program check:

- **Privilege violate** — The instruction is encountered while in problem state. The instruction is suppressed and a program check interrupt occurs with privilege violate set in the PSW.

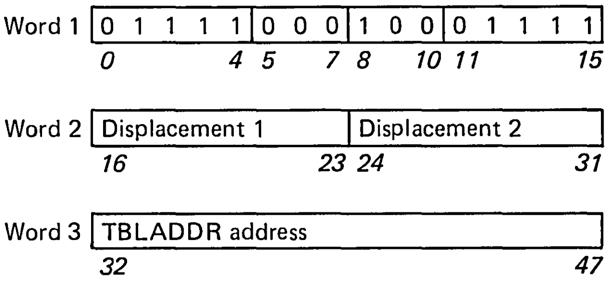
New Instructions

The processor has three additional instructions:

- Address Resolution with Indirect Branch (ARIB)
- Address Resolution with Indirect Branch—On (ARIBON)
- Address Resolution with Indirect Branch—Off (ARIBOFF).

ARIB Instruction Format

The ARIB instruction is a three-word instruction that performs the mapping actions. This instruction is intended to be used exclusively by the operating system. Mapping actions are the operation of fetching a command with an op code, operand 1, operand 2, and resolving the addressing modes used by those operands. The format of the instruction follows:



The instruction uses the values of displacement 1, displacement 2, and the address of the first word of the branch table TBLADDR for loading the ARIB instruction work registers, where:

- Displacement 1 is the index parameter displacement byte used with the control block designated by R2.

- Displacement 2 is the displacement byte parameter used with the control block designated by R2 to store R1.

- TBLADDR is the address of the first word of the branch-to table.

Register and Storage Conventions

The register conventions used by the mapping routine of the ARIB are as follows:

- Entry conditions:
  - R1 is the address of first word of the current command for ARIB.
  - R2 is the address of a control block.
- Exit conditions:
  - R1 and R2 are unchanged.
  - R3 is the resolved address of operand 1.
  - R4 is the resolved address of operand 2.
  - R5 is the contents of storage word pointed to by R1.
  - R6 is the TBLADDR index value that is defined as the command op code, multiplied by 2.
  - IAR is the contents of the storage location at the effective address TBLADDR, plus the contents of R6.
  - Storage location is the effective address of the contents of R2, plus displacement 2 of ARIB instruction word 2, containing the contents of R1.

Indicators: The indicators are not changed.

Program Checks: This instruction has the following program checks:

- **Invalid function** — The instruction encountered the ARIB mode condition off. The instruction is suppressed, and a program check interruption occurs, with invalid function set in the PSW.
- **Invalid storage address** — One or more words of the instruction or any of the effective addresses is outside the installed storage size of the system. The indirect branch does not occur, and the instruction is terminated. Some of the level registers may have changed.
- **Specification check** — The effective address or indirect address results in an even-byte boundary violation. Branching does not occur, and the instruction is terminated. Some of the level registers may have changed.

ARIBON Instruction Format

The ARIBON instruction is a two-word instruction that activates the address resolution and indirect branch feature by turning on an internal flag that allows execution of the ARIB instruction. This mode is only reset by an IPL, a system reset, a power-on reset, or the ARIBOFF instruction. The ARIBON instruction is intended to be used by the operating system, and it may be issued in either the problem state or the privilege state.

0 1 1 1 1	0 0 0	1 0 0	0 1 1 1 0
0 4	*5 7*	*8 10*	*11 15*

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
16 31

Word 2 bits 0–14 are reserved and must be set to 0. Bit 15 must be set to 1.

Indicators: The indicators are not changed.

ARIBOFF Instruction Format

The ARIBOFF instruction is a two-word instruction that deactivates the address resolution and indirect branch feature by turning off the ARIBON flag. This instruction is intended to be used by the operating system, and it may be issued in either the problem state or the privilege state.

0 1 1 1 1	0 0 0	1 0 0	0 1 1 1 0
0 4	*5 7*	*8 10*	*11 15*

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
16 31

Word 2 is set to all 0's.

Indicators: The indicators are not changed.

ARIB Execution

The required operations for the ARIB instruction are as follows:

Initialization: Prior to the invocation of the ARIB instruction, the operating system must issue the ARIBON instruction. For ordinary usage, this should be part of the IPL procedure.

Setup: The content of R1 is placed in the main storage location designated by the effective address of displacement 2 of the ARIB instruction word 2, plus the control block address in R2 and with OP2K to generate the physical storage address.

> **Note:** There is no use of this storage address during the ARIB instruction, and the R1 register is maintained intact.

Command: A high level command is fetched from the main storage location specified by the contents of R1, and placed into R5. The contents of R5 are defined as a command with the following definition.

I	X X X	OPR2	OPR1	Op code function
0	*1 3*	*4 5*	*6 7*	*8 15*

Bit	**Meaning**
0	Operand 2 type (1=constant, 0=address)
1–3	Not defined
4–5	Register flag for operand 2
6–7	Register flag for operand 1
8–15	Operation code function

The I-field of the op code indicates to the mapping routine that the second operand is an immediate value. The OPR2 and OPR1 fields are used to define the addressing mode used for each of the operands.

The op code function field is used by the mapping routine to generate the index value for displacement in the table "TBLADDR."

The 2-bit register flag for each operand is defined as follows:

Field value	**Meaning**
0	Register not specified
1	Operand 1 used in the form (d, index register 1)
2	Operand 2 used in the form (d, index register 2)
3	Operand 1 or operand 2 used explicitly

Operand Address Resolution: Operand 1 is fetched from the main storage location at the effective address specified by the contents of R1 plus 2, and placed into R3.

Operand 2 is fetched from the main storage location specified by the contents of R1 plus 4, and placed into R4. If operand 2 is a constant, R4 contains the value R1 plus 4, which is a pointer to that constant.

If operand 1 is an indexed value (parameter or index register), the setup routine adds the contents of the selected register (operand 1 or operand 2) to R3. If operand 2 is an indexed value (parameter, index register), the setup routine adds the contents of the selected register (operand 1 or operand 2) to R4.

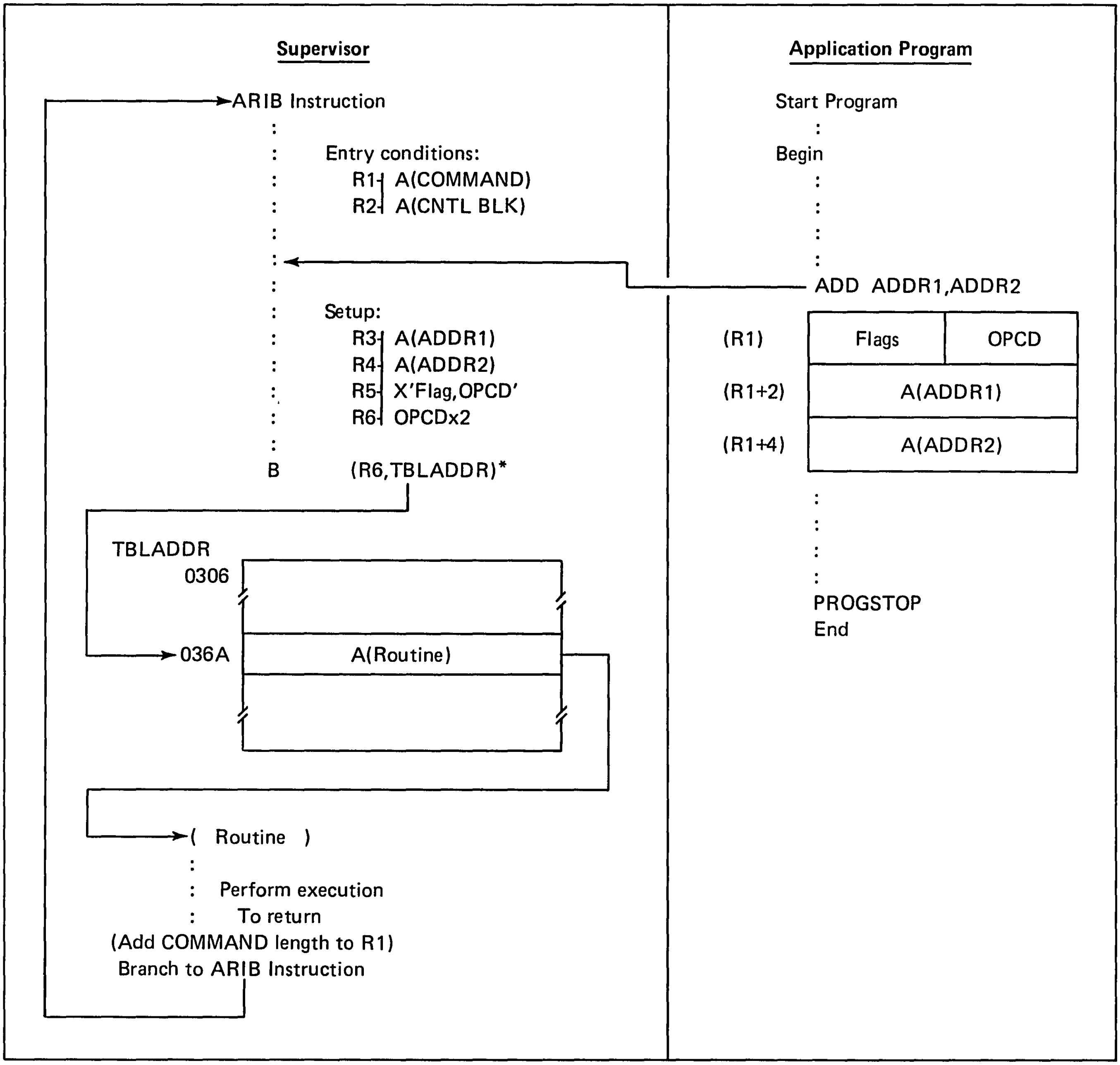
These are accomplished as follows: An effective address is generated by adding displacement 1 with the contents of level register R2. The effective address is used with OP2K to generate the physical address. The contents of main storage at the generated address contains an address which points to the first word of a pair of index values. Reference to this storage location is required whenever command word bits 4–5 or 6–7 are respectively non-zero. Another address is then formed by adding to these contents the bits 4–5, for operand 1, or bits 6–7, for operand 2, to the low-order address word bit positions, thus adding bit positions 13, 14, and 15 where bit 15 is 0. This address is then used to fetch the storage location that is added to the appropriate operand register, either R3 or R4. This represents the operand indexed addressing.

After setting up the registers with the proper addresses, the branch-to table address is resolved as follows:

Address Resolution: The address of the routine that processes the next instruction is located in the branch-to table at an offset that is equal to the command op code multipled by 2. Thus, add the TBLADDR base address to twice the product of the op code in hex.

Example:

The next instruction routine for the ARIB processing shown is calculated as follows:



Op code	Hex 32
Multiplied by 2	Hex 64
Plus TBLADDR base offset of	Hex 0306
Equals the offset into TBLADDR of the routine that processes the next instruction	Hex 036A

Using the content of R6, plus the table address (TBLADDR), as an effective address used with the ISK to generate the physical address, the content of that storage location is loaded into the level instruction address register (IAR). Thus, performing an indirect branch through the branch-to table to the proper next instruction routine completes the ARIB instruction execution.

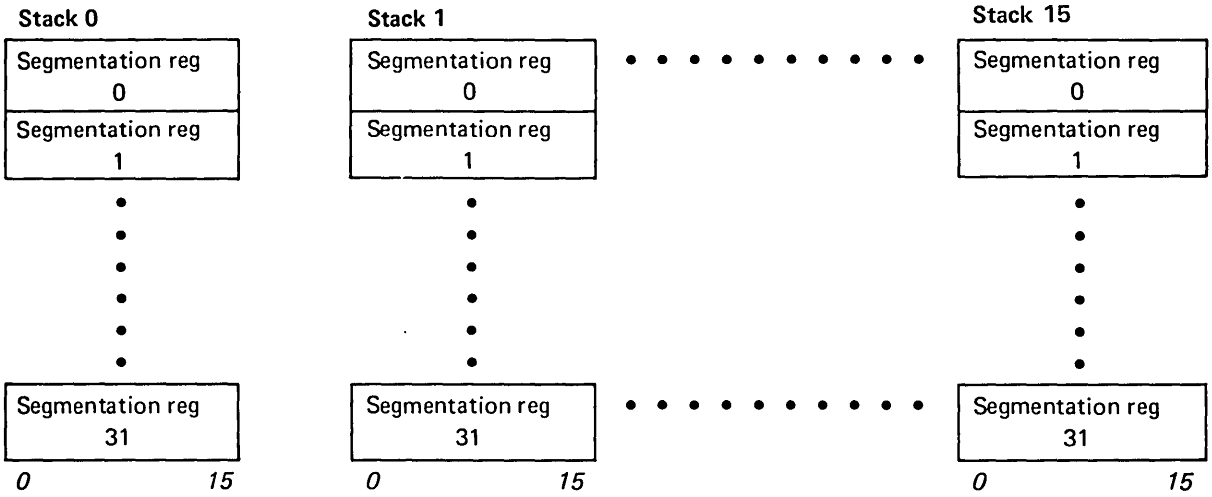
Chapter 2. Main Storage Addressing Using the Relocation Translator

The relocation translator and segmentation registers permit addressing of main storage locations beyond 64K bytes and provide a read-only type of storage protection. The first 64K bytes can be addressed directly when the translator is disabled; therefore, the translator must be enabled when main storage above 64K bytes is accessed.

Translator Description

The translator provides 16 stacks of 16-bit segmentation registers. The stacks are numbered 0–15 to correspond to the 16 possible values of the address keys. Each stack consists of 32 registers (0–31). In 3-bit mode, only stacks 0–7 (eight stacks) are used for translation, thereby allowing only 512K bytes of direct mappable storage.

Segmentation registers



The stacks of segmentation registers are under supervisory program control. Four privileged instructions are used with the relocation translator and segmentation registers.

- **Set Segmentation Register (SESR).** This instruction loads segmentation registers.

- **Copy Segmentation Register (CPSR).** This instruction allows the supervisor to inspect the contents of segmentation registers.

- **Enable (EN).** This instruction enables the relocation translator. Until the translator is enabled, 16-bit addressing is in effect for the low-order 64K bytes of storage. Any storage above 64K bytes is not accessible to the program until the translator is enabled.

- **Disable (DIS).** This instruction disables the relocation translator.

For further information about the preceding instructions, refer to their descriptions in chapter 1.

Storage Mapping

Mapping of main storage is achieved through the segmentation registers. Each segmentation register controls a 2K-byte segment of storage. The SESR instruction is used to load each segmentation register with the unique physical address of a 2K-byte segment of storage.

> **Note:** More than one segmentation register can be loaded with the same segment address. For example, stack 0, register 15 (associated with the supervisor address key of 0), can be loaded with the same number as stack 1, register 6. This arrangement allows the supervisor to address control blocks within a problem program even though the address key for the supervisor is different than the key for the problem program. Once loaded, each stack of segmentation registers contains a complete map of 64K bytes divided into 2K-byte physical segments.

Relocation Addressing

The relocation translator generates a physical address that allows any byte in storage to be addressed. Figure 2-1 shows an example of address translation. The letters in the following description correspond to the letters in Figure 2-1:

Ⓐ The active address key from the address key register selects a segmentation register stack. The address key pertains to the instruction being executed on the current priority level. Bit 12 is ignored in 3-bit mode.

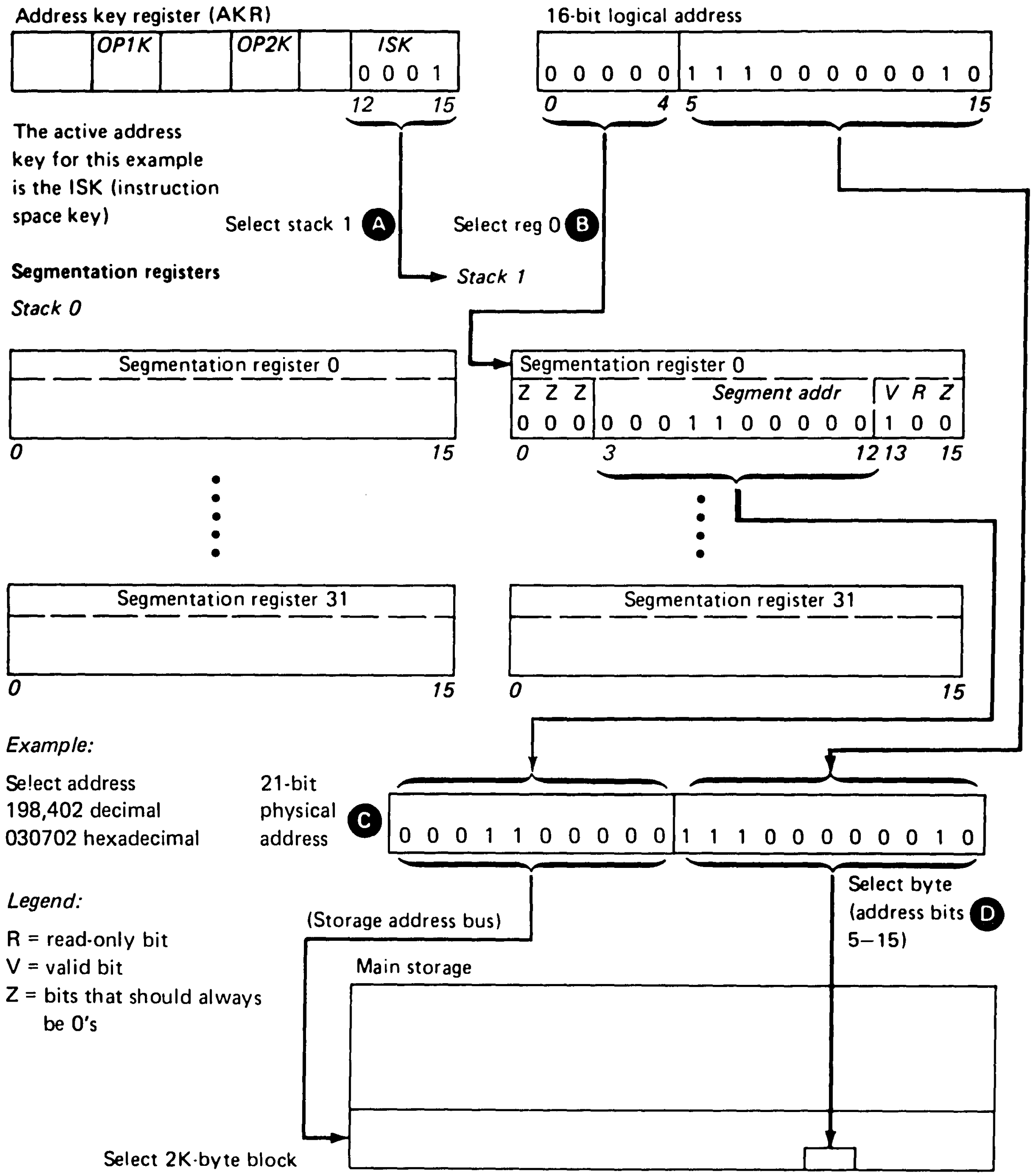
Ⓑ The five high-order bits (0–4) of the 16-bit address (generated for the instruction being executed) select a segmentation register within the stack selected in description Ⓐ. These bits define the logical segment.

Ⓒ The physical address is generated. The high-order bits are from the segmentation register; these bits specify the physical address of a 2K-byte segment of storage.

Segmentation Register Bit 13 — Valid Bit: When set to 1, this bit specifies that the contents of the segmentation register are valid; the segmentation register can be used to perform the translation. When bit 13 is a 0, the segmentation register cannot be used for translation (no access). If translation is attempted, a program-check interrupt occurs with invalid storage address set in the processor status word (PSW). (All valid bits are set to 0's after power is switched on.)

Segmentation Register Bit 14 — Read-Only Bit: When set to 1, this bit specifies that the block is read-only. If an attempt is made to write into storage using a segmentation register with the read-only bit set to 1, a program-check interrupt occurs with protect check set in the PSW. Storage is not changed. Bit 14 is ignored by a cycle-steal access or when the processor is in supervisor state.

Ⓓ The 11 low-order bits (5–15) of the physical address are the 11 low-order bits (5–15) of the 16-bit logical address (generated for the instruction being executed); these bits specify the byte address within the 2K-byte segment.



Example:

Select address
198,402 decimal
030702 hexadecimal

Legend:

R = read-only bit
V = valid bit
Z = bits that should always be 0's

Note: When the translator is disabled, address bits 0–15 only are used for main storage address selection.

Figure 2-1. Address Translation Example

I/O Storage Access Using the Relocation Translator

All storage access requests from I/O devices are translated by the same hardware that handles storage requests from the processor. The device control blocks (DCBs) must reside in the supervisor's address space; therefore, all I/O devices must use address key 0 to gain access to the DCBs and to store the individual residual status blocks. The address key of the process requiring a cycle-steal operation resides in a DCB. An I/O device presents this address key, along with a 16-bit logical address, to the relocation translator. This allows an I/O device to directly address the storage space for a particular process. The address key allows I/O storage protection to be established between address spaces, assuming that the supervisor ensures the integrity of the DCBs. In 4-bit translated mode, cycle-steal operations gain access to main storage through the I/O segmentation registers.

Status of Translator After Power Transitions and Resets

The translator is enabled by the Enable (EN) instruction, or by the PSW key of the programmer console, if installed. The translator is disabled by any of the following:

- Disable (DIS) instruction
- Power-on reset
- Check Restart key on programmer console
- Initial program load (IPL)
- System Reset key on programmer console.

All translator controls are reset when the translator is disabled.

Notes:

1. A machine-check interrupt does not disable the translator.
2. The segmentation registers are not reset when the translator is disabled.
3. The valid bits are all set to 0's when power is switched on.

Error-Recovery Considerations

Invalid Storage Address (ISA)

The invalid storage address bit (bit 1 of the PSW) is set to 1 by any one of the following:

- Storage access was attempted using a physical address greater than the physical storage size installed.
- Storage access was attempted with bit 13 (valid bit) of the segmentation register set to 0. This signifies that the contents of the segmentation register are invalid.

The specific nature of the invalid storage address can be resolved as follows:

- Store the segmentation register following the program-check interrupt.
- Test the value of bit 13 in the selected segmentation register. When set to 1, this bit specifies that the contents of the segmentation register are valid; the segmentation register can be used to perform the translation. When bit 13 is a 0, the segmentation register cannot be used for translation (no access). If translation is attempted, a program-check interrupt occurs with invalid storage address set in the processor status word (PSW).
- Ensure that the segment address does not exceed the limits of the physical processor storage installed.

Protect Check

When the translator is enabled, a program-check interrupt with protect check set in the PSW is caused by an attempt to write into storage, while in the problem state, using a segmentation register with bit 14 (read-only) set to 1.

Storage is not changed. Bit 14 is ignored by a cycle-steal access, or when in supervisor state.

Address Space Management

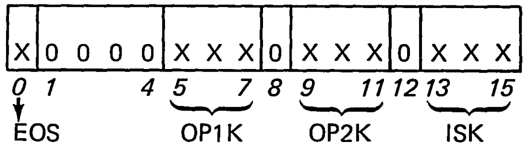
Active Address Key

Cycle-steal devices have a cycle-steal address key specified in their device control block.

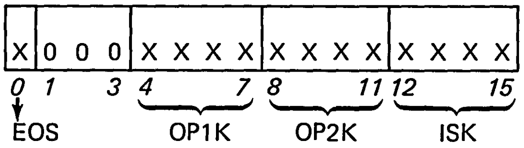
Any one of the four address keys (ISK, OP1K, OP2K, or cycle-steal address key) may be used during a storage access as the active address key. The address key in use (active) depends on the type of operation being performed at a specific instant in time. The active address key defines storage access through a particular block of segmentation registers.

Each priority level in the processor has an associated address key register (AKR) that contains an equate-operand-spaces (EOS) bit and three address keys (OP1K, OP2K, and ISK). In 3-bit mode, three bits are used in each address key; in 4-bit mode, four bits are used in each address key.

3-bit mode address key register (AKR)



4-bit mode address key register (AKR)



EOS *Equate operand spaces.* This bit, when set to 1, causes all data operands to use the OP2K address key. See "Equate Operand Spaces (EOS)" in this chapter.

OP1K *Operand 1 key.* These bits contain the binary-coded operand 1 address key, with bit 7 as the low-order bit.

OP2K *Operand 2 key.* These bits contain the binary-coded operand 2 address key, with bit 11 as the low-order bit.

ISK *Instruction space key.* These bits contain the binary-coded instruction-space address key, with bit 15 as the low-order bit.

Equate Operand Spaces (EOS)

The equate operand spaces bit (bit 0) in the address key register controls the use of the OP1K address key.

When the EOS bit is set to 1 (enabled), all processor data fetches use a single address space defined by the OP2K address key. The OP1K is ignored, but not changed, and all normal OP1K operations use OP2K as an active key. When the EOS bit is set to 0 (disabled), the OP1K address key functions in a normal manner.

Equate operand spaces (EOS) may be enabled by an Enable (EN) instruction, a Set Level Block (SELB) instruction, or a Set Address Key Register (SEAKR) instruction. EOS may be disabled by a Disable (DIS) instruction, a Set Level Block (SELB) instruction, or a Set Address Key Register (SEAKR) instruction. The EOS is also disabled by a priority interrupt or a class interrupt. These instructions are described in chapter 1 and in *IBM Series/1 Principles of Operation*, GA34-0152.

Address Space

When the relocation translator is enabled, an address key defines a specific address space where:

- The address space is a range of logically contiguous storage.

- The address space is accessible by the effective address without operating system intervention (the address space is not greater than 64K bytes).

All instruction fetches use the address space defined by the instruction space key (ISK). For storage-to-storage instructions, all reads and writes for data operand 1 use the address space defined by the OP1K, assuming that the EOS bit is a 0. All other storage data accesses, reads, and writes use the address space defined by the OP2K, excluding branch and jump instructions.

Examples:

ISK=OP1K=OP2K. For instruction processing, all storage accesses occur within the same address space.

ISK≠OP1K, OP1K=OP2K. Instruction fetches occur in the ISK address space. Data access occurs in the OP2K address space.

ISK≠OP1K, OP1K≠OP2K. Refer to Figure 2-2 for this example.

I/O operations that access main storage also use an address key. Cycle-steal operations (read or write) use the cycle-steal address key specified within the device control block. An address key of 0 is used when the device fetches the device control block. Direct program control (DPC) operations that write data to storage use the OP2K address key.

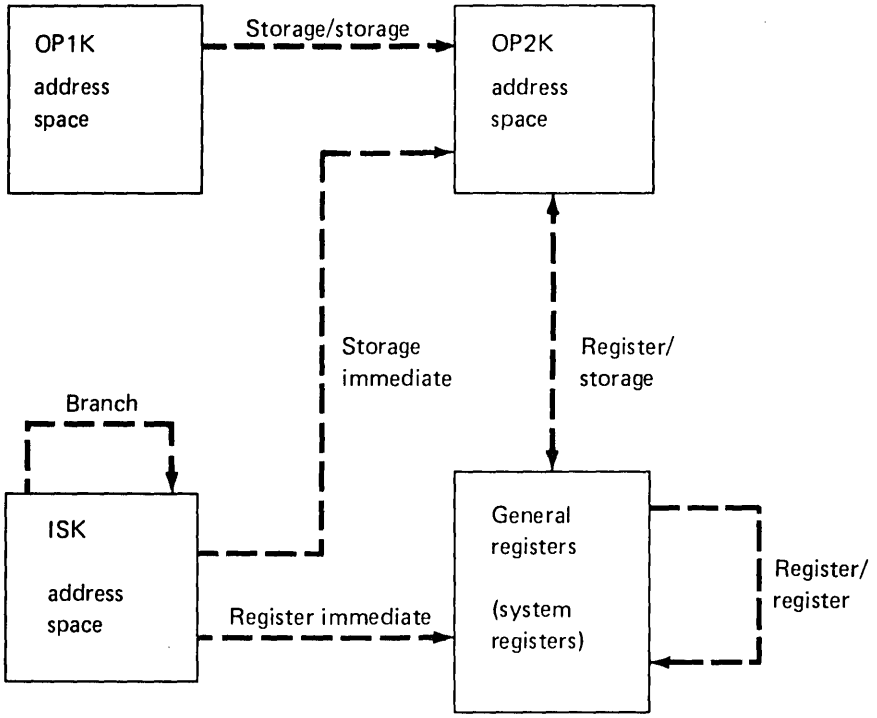
Other defined uses of the address key register are as follows:

- All indirect access for branching uses the ISK.

- Effective-address generation occurs in the address space of the particular data operand. The appended words in the instruction are accessed by the ISK.

- Storage access from the console is defined by the SAR address key. Stop-on-address is based on the Stop On Address key when the translator is enabled.

- System reset and IPL set all address keys and the EOS bit to 0's.



Assembler syntax for address spaces (see Appendix A)

ISK	OP1K	OP2K	Example instructions	
	addr5	addr4	AW	addr5,addr4
	(reg)	(reg)	MVFD	(reg),(reg)
Bits 12-15 of AKR			MVBI	byte,reg
Bits 12-15 of AKR			B	longaddr*

*Indirect addressing.

Notes:

1. OP1K is only used for the source operand in storage-to-storage operations.
2. OP2K is used for storage data access in all other operations (excluding branch/jump).
3. ISK (bits 13-15 of the AKR) is used for instruction fetch and branch/jump operations.
4. In 3-bit mode, bit-12 of the AKR is ignored.

Figure 2-2. Data Movement in Address Spaces When ISK≠ OP1K, OP1K≠ OP2K

Address Key Values After Interrupts

When priority or class interrupts occur, certain values are set in the address keys of the affected AKR. These values anticipate the address spaces that the programmer might need for interrupt processing. Figure 2-3 shows the resulting AKR values for each type of interrupt:

Interrupt	**EOS**	**OP1K**	**OP2K**	**ISK**
Priority	0	0	0	0
Supervisor call	0	Note 1	0	0
Machine check	0	Note 2	0	0
Program check	0	Note 2	0	0
Soft-exception trap	0	Note 1	0	0
Trace	0	Note 3	0	0
Console	0	0	0	0
Power/thermal warning	0	0	0	0

Notes:

1. OP1K is set to the preceding key contained in OP2K.
2. OP1K is set to the last active processor address key.
3. OP1K is set to the preceding key contained in the ISK.

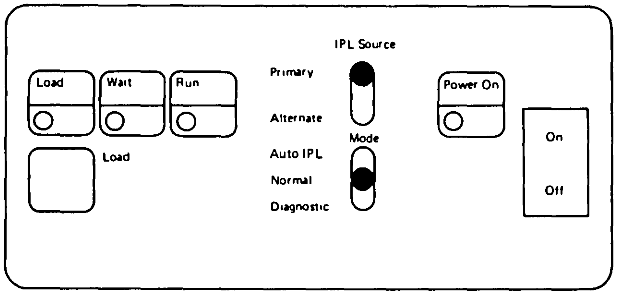
Figure 2-3. Resulting AKR Values

All interrupt service routines reside in address space 0; therefore, the ISK and OP2K are set to 0's when an interrupt occurs. Necessary information for processing a specific interrupt may reside in an address space other than 0. The address key related to the particular interrupt is placed in OP1K. The OP1K is set in anticipation of a storage-to-storage move of information from the interrupting address space to address space 0.

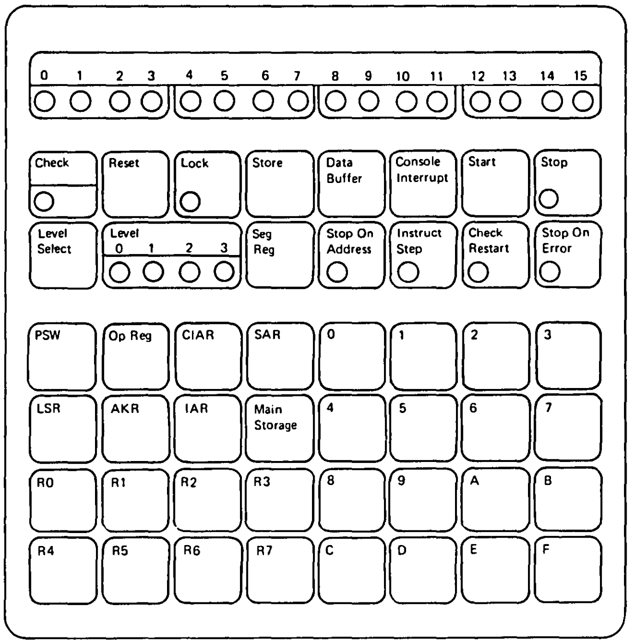
> **Note:** Class interrupts cause a hardware-controlled storing of a level status block. This operation uses address key 0.

Chapter 3. Console

The basic console is a standard standard feature of the processor; the programmer console is an optional feature.



Basic console



Programmer console

The basic console is intended primarily for those systems that are totally dedicated to a particular application, where operator intervention is not needed during the execution of the application.

The programmer console is intended for operator-oriented systems where various programs are entered and executed. This type of environment requires a more versatile console arrangement for program and machine problem determination, and for manual alteration of data and programs in storage.

Basic Console

Each 4956 comes equipped with a basic console, which provides the following:

- Power On/Off switch for the processor unit
- IPL Source switch to select a primary or alternate IPL device
- Load key for initial program load (IPL)
- Mode switch to select: Auto IPL, Normal, or Diagnostic mode
- Load, Wait, Run, and Power On indicators.

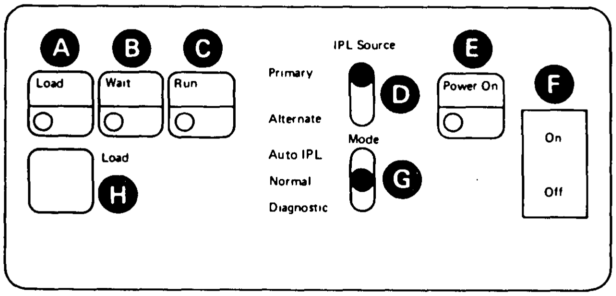
Ⓕ **Power On/Off:** When this switch is set to the On position, power is applied to the processor unit. After all power levels are up, the Power On indicator is turned on. When this switch is set to the Off position, power is removed from the processor unit and the Power On indicator is turned off.

Ⓓ **IPL Source:** This switch selects the I/O device to be used for program loading. In the Primary position, the device that was pre-wired as the primary IPL device is selected. In the Alternate position, the device that was pre-wired as the alternate IPL device is selected.

Ⓗ **Load:** Pressing this key causes a system reset, and the initial program load (IPL) sequence is started. The Load indicator is turned on and remains on until the IPL sequence is completed. When the IPL sequence is completed, instruction execution begins at location 0 on priority level 0.

Ⓖ **Mode:** This switch has the following positions:

- Auto IPL—In this position, an IPL is initiated after a successful power-on sequence. Bit 13 of the PSW is set to indicate to the software that an automatic IPL was performed. In this mode, Stop instructions are treated as no-ops.
- Normal—In this position, Stop instructions are treated as no-ops.
- Diagnostic—This position has no function without the programmer console. This position places the processor in diagnostic mode if the programmer console is attached. When the processor is in diagnostic mode, Stop instructions cause the processor to enter the stop state.



Indicators

Ⓐ **Load:** On when the machine is performing an initial program load (IPL).

Ⓑ **Wait:** On when an instruction that exits the active level has been executed and no other priority interrupts or levels are pending.

Ⓒ **Run:** On when the machine is executing instructions.

Ⓔ **Power On:** On when the proper power levels are available to the system.

Programmer Console

The programmer console is an optional feature that can be ordered with the 4956 or field-installed at a later date. The programmer console provides the following:

- Start and stop of the processor.
- Ability to display or alter any storage location.
- System reset.
- Selection of any one of the four interrupt levels for the purpose of displaying or altering data.
- Displaying or altering of the storage address register (SAR), instruction address register (IAR), SAR address key register (AKR), stop-on-address address key register (AKR), level address key register (AKR), segmentation registers, console data buffer, or any general purpose register.
- Displaying, but not altering, the level status register (LSR), current instruction address register (CIAR), op register, or processor status word (PSW). Note that the following bits of the PSW and LSR may be altered: PSW bit 14 (translator enabled), PSW bit 7 (4–bit mode), LSR bit 8 (supervisor state), and LSR bit 11 (summary mask).
- Stop on address.
- Stop on error.
- Instruction stepping.
- Check restart.
- Request for a console interrupt.
- Check indicator. The Check indicator is a light emitting diode (LED) that lights when a machine check or program check class interrupt occurs.
- Lock console.
- CE mode. The CE mode is used to display the error log.

The programmer console is touch-sensitive, with an audio-tone generator providing an audio response tone whenever a key is pressed and the information has been accepted and serviced by the processor.

Console Display

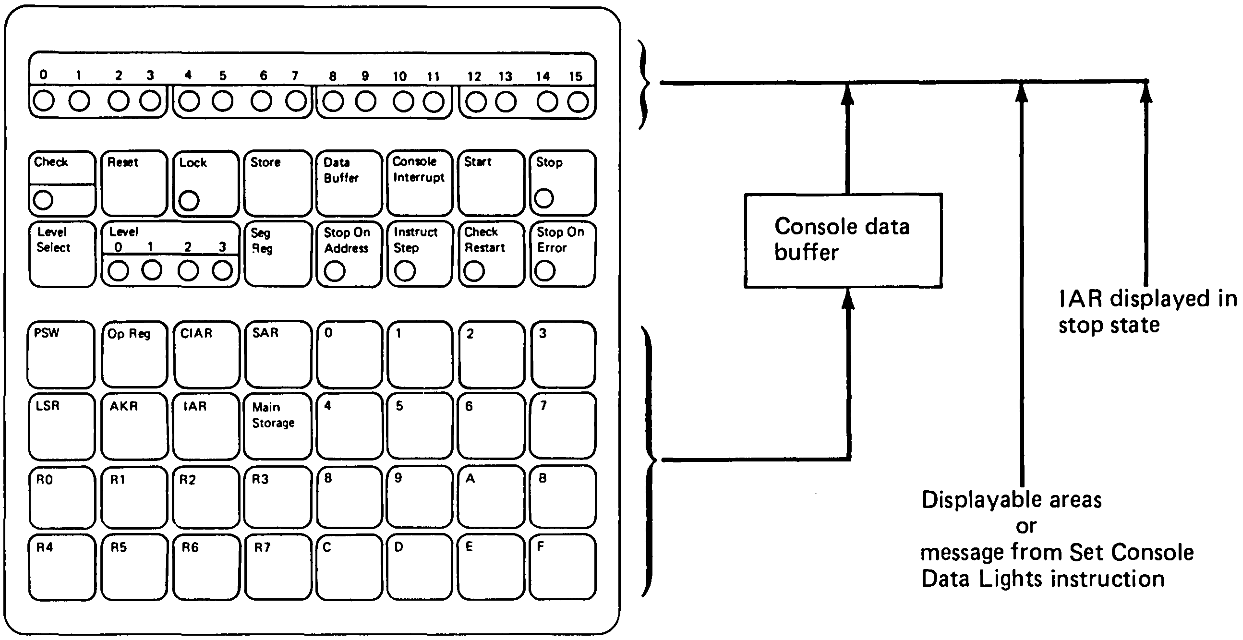
Run or Wait State

When the processor is in run or wait state, the console data buffer is displayed in the data display indicators. An exception to this is when a Set Console Data Lights (SECON) instruction writes a message to the data lights and does not change the buffer. When the Data Buffer key is pressed, the console data buffer is again displayed in the indicators.

When the console data buffer is being displayed, the console data buffer and the display are changed by entering new data with the data entry keys.

Stop State

When the processor enters stop state, the IAR is displayed in the data display indicators. Any system resource that has a corresponding select key on the console can be displayed. For example, the console data buffer can be displayed by pressing the Data Buffer key.



Power-On Reset

After a successful power-on reset, the data display indicators are set on, and the Stop indicator is set on (if the Mode switch is not positioned for Auto IPL).

Indicators

Ⓐ **Data Display:** When the processor is in run state, the console data buffer is displayed in the data display indicators.

The Set Console Data Lights (SECON) instruction can write a message to the data display.

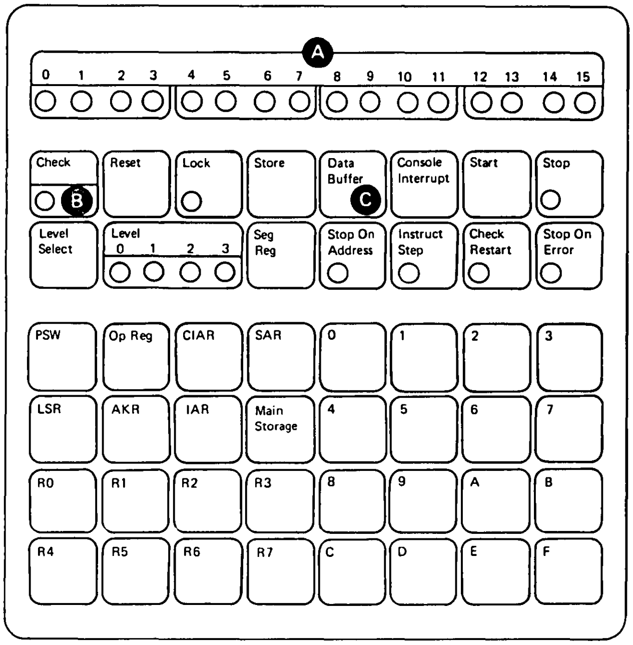
When the processor enters stop state, the IAR is displayed until another system resource is selected.

To display the contents of the console data buffer after a system resource has been displayed, press the Data Buffer key Ⓒ.

Ⓑ **Check:** On when a machine-check or program-check has been recognized. The Check indicator is turned off by:

- Clearing the check condition.
  - Reset key.
  - Load key.
  - Executing a Copy Processor Status and Reset (CPPSR) instruction. This instruction resets bits 0–6 and bits 8–12 of the PSW.
- Pressing any console key while in the stop state. The check condition is not cleared unless the Reset key or the Load key is pressed.

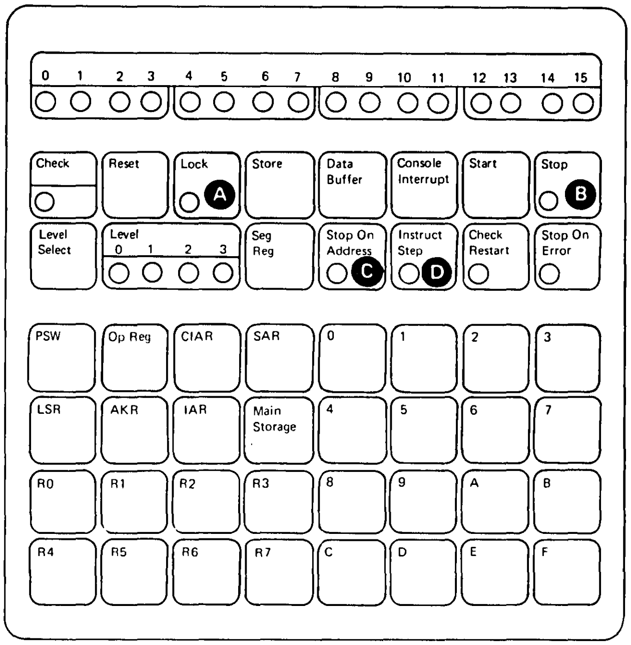
While in the stop state, the Check indicator is used to indicate main storage parity errors or invalid storage addresses during display operations. Refer to "Displaying Main Storage Locations" in this chapter.



Combination Keys/Indicators

There are six combination keys/indicators:

- Lock
- Stop
- Stop On Address
- Instruct Step
- Check Restart
- Stop On Error.



A **Lock:** Pressing the Lock key first (Lock LED begins flashing), then pressing four hex keys and the Store key locks the console. A locked console is indicated by an illuminated indicator on the Lock key. The data LEDs are automatically set to the previous value. Displays or alterations cannot be performed with the programmer console keys while in the Lock mode. The console remains locked until the same sequence of hex keys that locked the console is repeated and then followed by pressing the Store key.

The only data displayed during the lock mode is data set by the program or data displayed during a maintenance procedure (CE) mode.

Lock mode is automatically reset during a power-on sequence. If the console is locked and an auto IPL occurs after a power failure, the console will not be locked after the power-on.

If the console is locked in the stop mode, the only active switches are Lock, Store, and the hex keys. At this time, the run mode cannot be entered; therefore, for a normal lock function, lock mode should only be set during the run mode.

CE Mode: The CE mode may be used (to allow the user to display the error log) by the following:

1. Press the Lock key; the Lock LED flashes.

2. Press the hex keys in the sequence : C, E, 0, 0.

3. Press the Store key; the most recent entry in the error log is indicated by the display LEDs.

4. Press the Lock key once; the previous entry in the error log is indicated by the display LEDs. The Lock key may now be pressed as many times as desired. Each time the Lock key is pressed, the next previous error log entry appears on the display LEDs. (Refer to Appendix C for a description of the error log.)

   Other keys may be pressed between subsequent operations of the Lock key. CE mode is exited when all 64 entries have been displayed, or when the Store key is pressed immediately after the Lock key is pressed.

   Upon entering a lock/unlock/CE mode sequence, the Lock LED flashes. The SECON instruction is disabled until the lock/unlock/CE mode sequence is terminated. The console data LEDs then assume their former value, their value upon entering stop state if in stop state, or the last value sent to them if SECON instructions have occurred.

B **Stop:** This indicator is on when the processor is in the stop state. Stop state is entered in the following ways:

- By pressing the Stop key.

  - In run state, the current instruction is completed.

  - In wait state, stop state is entered directly.

  - In stop state, the contents of the instruction address register (IAR) prior to entering the present stop state are restored to the IAR and displayed in the data display indicators. The level that was active upon entering stop state is reselected (becomes active).

- By execution of the Stop instruction (diagnostic mode only).

- When an address compare occurs in stop-on-address mode.

- When an error occurs in stop-on-error mode.

- By pressing the Reset key.

- When a power-on reset occurs.

- By selecting instruction-step mode while in run state.

The Stop On Address key and the Instruct Step key are mutually exclusive. When one is pressed, the other is reset if it is on.

Ⓒ **Stop On Address:** Pressing this key places the processor in stop-on-address (SOA) mode and turns on the Stop On Address indicator. Pressing this key a second time resets stop-on-address mode and turns off the indicator.

Ⓓ **Instruct Step:** Pressing this key places the processor in instruction-step mode and turns on the Instruct Step indicator. Pressing this key a second time resets instruction step mode and turns off the indicator.

If the processor is in run or wait state, pressing this key causes the processor to enter stop state. Pressing the Instruct Step key a second time resets instruction-step mode; the processor remains in stop state.

To operate in instruction step mode:

1. Key the desired starting address and store into the IAR.

2. Press the Instruct Step key.

3. Press the Start key. The instruction located at the selected address is executed, and the processor returns to stop state. The IAR is updated to the next instruction address; this address is displayed in the data display indicators.

Each time the Start key is pressed, one instruction is executed and the IAR is updated to the next instruction address.

> **Note:** Priority and class interrupts are not inhibited during execution of the instruction.

Stop-On-Address Mode

The processor must be in stop state to set the compare address.

Stop On Address (Relocation Translator Disabled)

1. Press the Stop On Address key.

   Contents of the stop-on-address register are indicated by the display LEDs.

2. Enter the selected stop-on-address address by pressing the hex entry keys for a four-digit hex address.

3. Press the Store key.

   Contents of the updated stop-on-address register are indicated by the display LEDs.

4. Press the Start key.

   Execution begins at the current IAR address on the level that was active prior to entering the stop state.

   When the selected address is loaded into the IAR, the processor enters the stop state (IAR stop only). When the compare occurs, the stop state is entered with the IAR displayed in the data display indicators.

Stop On Address (Relocation Translator Enabled)

1. Press the Stop On Address key.

   Contents of the stop-on-address (SOA) register are indicated by the display LEDs.

2. Press the AKR (address key register) key.

   Contents of the stop-on-address address key register are displayed.

3. Enter the desired address key by pressing one hex entry key for a digit value (hex 0 through 7) for 3–bit mode, or (hex 0 through F) for 4–bit mode.

4. Press the Store key.

   Contents of the updated stop-on-address key register are displayed.

5. Press the Stop On Address key.

   Contents of the stop-on-address register are indicated by the display LEDs.

6. Enter the selected compare address by pressing the hex entry keys for a four-digit hex address.

7. Press the Store key.

   Contents of the updated stop-on-address register are indicated by the display LEDs.

   The selected stop-on-address key register and stop-on-address register are used to compute a 21–bit physical address. Whenever the value in the segmentation register is changed, the physical address is recomputed.

   Note: The contents of the stop-on-address key register and the stop-on-address register may be displayed on the console; however, the 21–bit physical address cannot be displayed.

8. Press the Stop On Address key.

   The processor is now in stop-on-address mode.

9. Press the Start key.

   Execution begins at the current IAR address on the level that was active prior to entering the stop state.

   When the selected physical address is computed using the SAR and the active address key, the processor enters the stop state. The logical IAR is indicated by the display LEDs. The stop on address only stops on an instruction address.

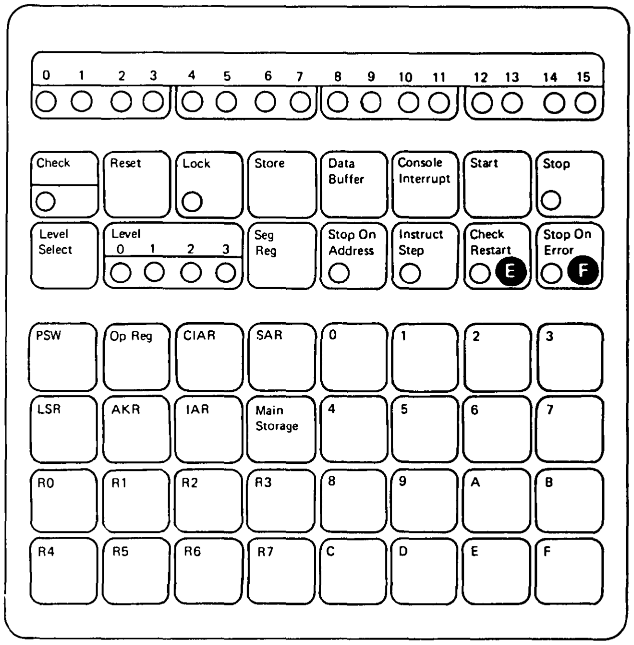
The Check Restart key and the Stop On Error key are mutually exclusive. When one is pressed, the other is reset if it is on.

E **Check Restart:** Pressing this key places the processor in check restart mode. While in this mode, a program-check, machine-check, or power/thermal-warning class interrupt causes the processor to be reset and execution to restart at address 0 on level 0.

> **Note:** The power/thermal-warning stop-on-error condition is controlled by the summary mask.

F **Stop On Error:** Pressing this key places the processor in stop-on-error mode. Any program-check, machine-check, or power/thermal-warning class interrupt causes the processor to enter stop state. To determine the cause of the error, display the PSW. To restart the processor, press the Reset key and then the Start key. Pressing only the Start key allows the processor to proceed with the class interrupt as if stop mode had not occurred. Note that the Check indicator may have been turned off while in stop state. After the class interrupt routine is completed, control may be returned to the instruction that caused the error and an attempt to reexecute the instruction may be made. Some instructions are not reexecutable because operand registers or storage locations were changed before the instruction was terminated (because of the initial error). In these cases, the operator must be familiar with the program because manual restoration of affected locations must be made before restart is attempted.

> **Note:** The power/thermal-warning class interrupt is controlled by the summary mask.



Keys and Switches

Ⓐ **Reset:** This key initiates a system reset that performs the following functions:

- IAR on level 0 set to 0
- AKR on level 0 set to 0
- Interrupt mask set to all levels enabled
- LSR on level 0—indicators set to 0's, summary mask enabled, supervisor state and in-process flag turned on, trace disabled
- LSRs for levels 1–3 set to 0's
- PSW bits 0–12 and 14 set to 0's (bit 14 set to 0 indicates translator disabled); bits 13 and 15 retain their state prior to system reset
- SAR set to 0
- CIAR set to 0
- Console display LEDs are turned off
- Clock class interrupts are disabled
- Error logging set to the enabled state
- Set to 3–bit mode.

After the system reset is completed, the processor is placed in the stop state with the Stop indicator on.

The following resources are not affected by system reset:

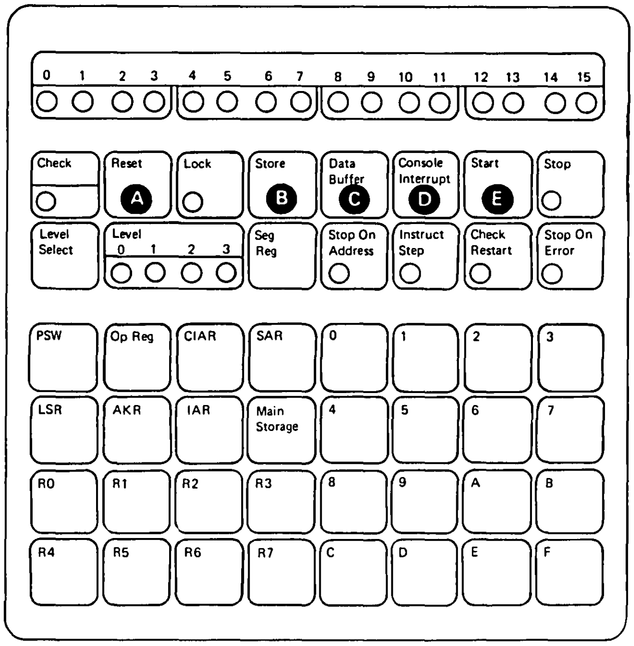
- General registers (all levels)
- IARs (levels 1–3)
- AKRs (levels 1–3)
- Main storage
- Console data buffer
- Segmentation registers
- Stop-on-address register
- Clock
- Comparator.

Ⓑ **Store:** This key is effective only when the processor is in stop state. Pressing this key causes the last data entry to be stored in the last selected resource.

Ⓒ **Data Buffer:** Pressing this key causes the console data buffer to be selected. The contents of the console data buffer are displayed in the data display indicators.

Ⓓ **Console Interrupt:** The effect of this key depends on the state of the processor. If the processor is in the stop or load state, this key has no effect. If the processor is in the run or wait state and the summary mask is enabled prior to the key action, a console-class interrupt occurs. The audio-response tone is generated when the interrupt is processed.

Ⓔ **Start:** This key is effective in stop state only. Stop state is exited and the processor resumes execution at the address in the IAR on the current level. If stop state was entered from system reset, execution begins at address 0, level 0. If stop state was entered from wait state, the processor returns to wait state.



Ⓗ **PSW:** Pressing this key selects the processor status word. The contents of the PSW are displayed in the data display indicators. Only PSW bit 14 (translator enabled) and PSW bit−7 (4−bit key enable) can be stored into the PSW from the programmer console.

Ⓙ **Op Reg:** Pressing this key selects the op register and displays the contents in the data display indicators. Data cannot be stored into the op register from the console.

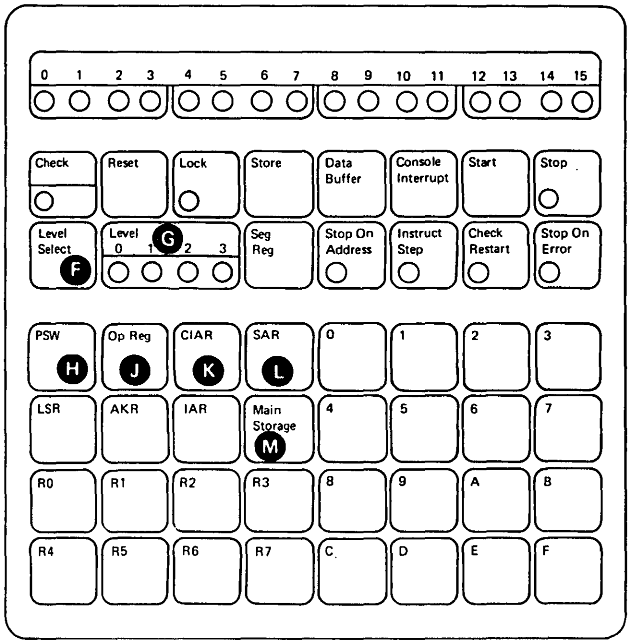
Ⓚ **CIAR:** Pressing this key, after entering stop state, causes the address of the instruction just executed to be displayed. Data cannot be stored into the CIAR from the console.

Ⓛ **SAR:** Pressing this key, while in stop state, displays the contents of the storage address register. An address can be stored into the SAR to address main storage or the segmentation registers for display or store operations. Bit 15 of the SAR cannot be set from the console.

Ⓜ **Main Storage:** Pressing this key selects main storage as the facility to be accessed by the console. When this key is pressed, the contents of the main storage location addressed by the SAR are displayed in the data display indicators. Procedures for displaying and storing main storage are described in subsequent paragraphs in this chapter.

Ⓕ **Level Select:** In the stop state, the Level-Select key should be pressed first, before selecting a new level. The desired level may then be selected by pressing either the 0, 1, 2, or 3 hex key.

The current active level (Level 0, 1, 2, or 3) is always displayed by one of the four level indicators at Ⓖ.



Level-Dependent Keys

The following keys select registers that are duplicated in hardware for each of the four interrupt levels:

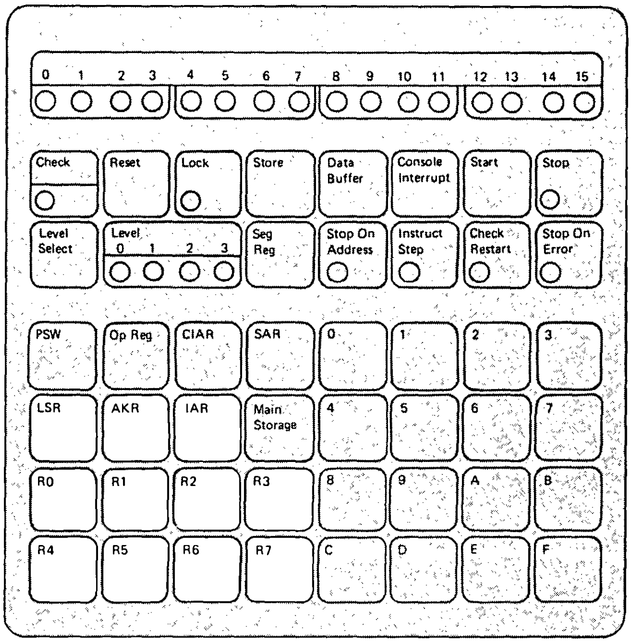
- LSR
- AKR
- IAR
- R0–R7 (General purpose registers 0–7).

Pressing any of these keys, once a level has been selected, causes the contents of that register to be displayed in the data display indicators.

The level status register (LSR) is displayable only, except bits 8 (supervisor state) and 11 (summary mask) can be stored into this register.

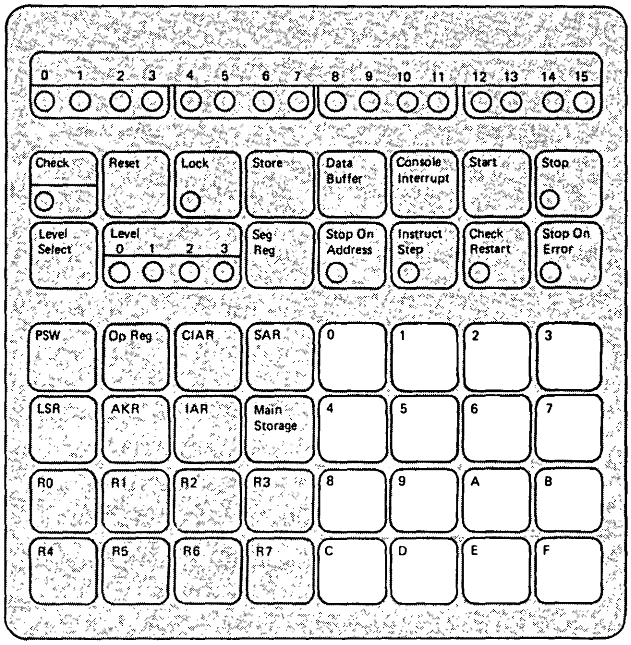
To display an AKR for a given level, enter the desired level, and then press the AKR key. The *level* AKR, bits 0, 5–7, 9–11, and 13–15 (EOS, OP1K, OP2K, and ISK) are displayed in the data display indicators for 3–bit mode. For 4–bit mode, EDS, OP1K, OP2K, and ISK are displayed in bits 0, 4–7, 8–11, and 12–15 of the data display indicators.

To display SAR AKR, first press SAR, then press AKR. To display the stop on address AKR, first press the Stop On Address key, then press AKR. To display CIAR AKR, first press CIAR, then press AKR (three bits, ISK). An AKR store is accomplished by first displaying the level AKR, then entering four hexadecimal digits, followed by pressing the Store key. When the Store key is pressed, the new level AKR is displayed. After the SOA AKR, or the SAR AKR is displayed, enter one hexadecimal digit and press the Store key. The CIAR AKR is displayable only.



Data Entry Keys

The 16 data entry keys are used to enter data into a selected resource, such as main storage or a general register. When data is entered, it is shifted through the indicators, as shown in the following example:



Example: Data to be entered: F3A8

Action	Data display indicators															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Press data entry key F	○	○	○	○	○	○	○	○	○	○	○	○	●	●	●	●
Press data entry key 3	○	○	○	○	○	○	○	○	●	●	●	●	○	○	●	●
Press data entry key A	○	○	○	○	●	●	●	●	○	○	●	●	●	○	●	○
Press data entry key 8	●	●	●	●	○	○	●	●	●	○	●	○	●	○	○	○

Legend:

● — Indicator on

○ — Indicator off

Displaying Registers

The processor must be in stop state.

1. Select the proper level by first pressing the Level Select key Ⓒ, then the appropriate 0, 1, 2, or 3 hex data key.

   The contents of any register associated with the selected level can now be displayed by pressing a register key.

2. Press the desired register key. The contents of that register are displayed in the data display indicators Ⓐ.

Storing Into Registers

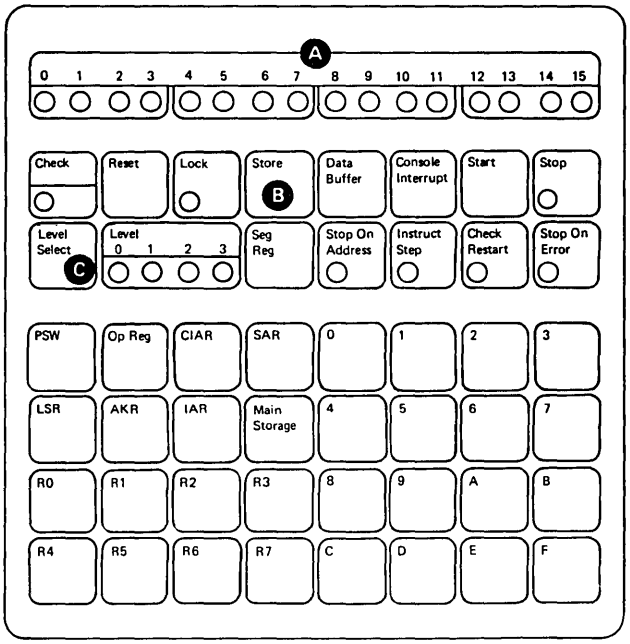
The processor must be in stop state.

1. Select the proper level by pressing the Level Select key Ⓒ, then the appropriate 0, 1, 2, or 3 hex data key.

2. Press the key for the register where data is to be stored. The contents of that register are displayed in the data display indicators Ⓐ.

3. Key in the data that is to be stored. This data is displayed in the data display indicators Ⓐ.

4. Press the Store key Ⓑ. The data that is displayed is stored into the selected register.



Displaying Segmentation Registers

The address relocation translator provides 16 stacks (0–15) of 32 segmentation registers (0–31) in each stack, for a total of 512 segmentation registers. Refer to "Relocation Addressing" in Chapter 2. In addition; for I/O segmentation registers, the address relocation translator provides eight stacks (0–7) of 32 segmentation registers (0–31) for a total of 256 segmentation registers.

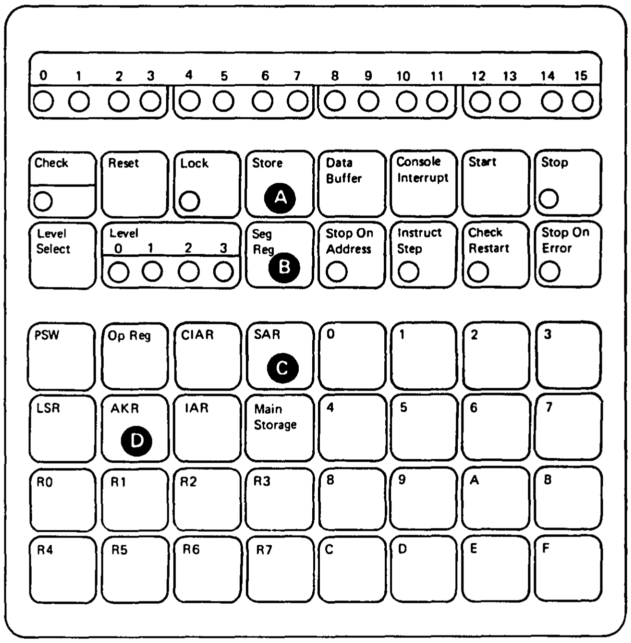
The processor must be in the stop state.

1. Press the SAR key Ⓒ. The contents of the SAR are displayed in the data display indicators.

2. Key in a hexadecimal four-digit number with the five high-order bits equal to the binary address (bits 0–31) of the desired segmentation register.

3. Press the Store key Ⓐ. The address is stored in SAR.

4. Press the SAR key Ⓒ. The selected address is displayed in the data display indicators.



5. Press the AKR key Ⓓ. The contents of the SAR address key register (AKR) are displayed in the data display indicators.

6. Key in one hexadecimal character to select the desired segmentation stack (0–F).

7. To select the I/O segmentation stack, key in hexadecimal 008X, where X is the desired stack (0–7).

8. Press the Store key Ⓐ. The value is stored in the SAR AKR.

9. Press the Seg Reg key Ⓑ. The contents of the selected segmentation register (defined by the five high-order bits of the SAR and the three/four SAR AKR bits) are displayed in the data display indicators.

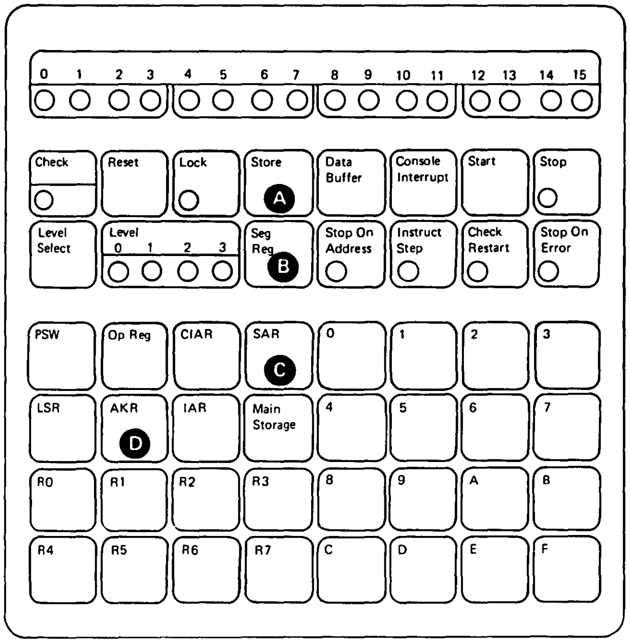
Note: Each time the Seg Reg key is pressed, the segmentation-selection address is incremented by 1 until the last segmentation register in the stack is selected. Then, the segmentation-selection address wraps from 31 to 0. When the segmentation-selection address wraps from 31 to 0, the SAR AKR is incremented by 1 (a new segmentation-register stack is selected); the new segmentation-register contents are displayed in the data display indicators. When all segmentation register stacks have been selected, the SAR AKR value then wraps from F (decimal 15) to 0, or 7 to 0 for I/O stacks.

Storing Into a Segmentation Register

The address relocation translator provides 16 stacks (0–15) of 32 segmentation registers (0–31) for a total of 512 segmentation registers. Refer to "Relocation Addressing" in Chapter 2. In addition, for I/O segmentation registers, the address relocation translator provides eight stacks (0–7) of 32 segmentation registers (0–31) for a total of 256 segmentation registers.

The processor must be in the stop state.

1. Press the SAR key Ⓒ. The contents of the SAR are displayed in the data display indicators.



2. Key in the value that selects the desired segmentation register within a stack (four hex characters entered with the data entry keys).

3. Press the Store key Ⓐ. The selected address is stored in the SAR.

4. Press the SAR key Ⓒ. The selected address is displayed in the data display indicators.

5. Press the AKR key Ⓓ. The contents of the SAR address key register (AKR) are displayed in the data display indicators.

6. Key in one hex character with a data entry key (any value from 0 through F, which is the new address key that selects a segmentation-register stack). This character is displayed in bits 12–15 of the data display indicators.

7. To select the I/O segmentation stack, key in hexadecimal 008X, where X is the desired stack (0–7).

8. Press the Store key Ⓐ. The contents of the SAR address key register (AKR) are updated to the value entered from the data entry keys.

9. Press the Seg Reg key Ⓑ. The contents of the selected segmentation register (defined by the five high-order bits of the SAR and the three/four SAR AKR bits) are displayed in the data display indicators.

10. Key in the value (four hex characters entered at the data entry keys) that provide both the desired ten high-order bits of the 21-bit physical main storage address (select a 2K-byte block of main storage) and that contain the correct value for the valid bit and the read only bit.

11. Press the Store key Ⓐ. The selected segmentation register is updated to the value in the data display indicators.

    Note: Each time the Store key is pressed, the last value keyed is entered into the selected segmentation register and the segmentation selection address is incremented by 1 until the last segmentation register in the stack is selected. Then, the segmentation selection address wraps from 31 to 0. When the segmentation selection address wraps from 31 to 0, the SAR AKR is incremented by 1 (a new segmentation-register stack is selected); the new segmentation-register contents are displayed in the data display indicators. When all segmentation register stacks have been selected, the SAR AKR value then wraps from F to zero, or seven to zero for I/O stacks.

The segmentation registers can be written into by the Set Segmentation Register (SESR) instruction and can be displayed by the Copy Segmentation Register (CPSR) instruction. Refer to *IBM Series/1 Principles of Operation*, GA34-0152, for additional information.

Displaying Main Storage Locations

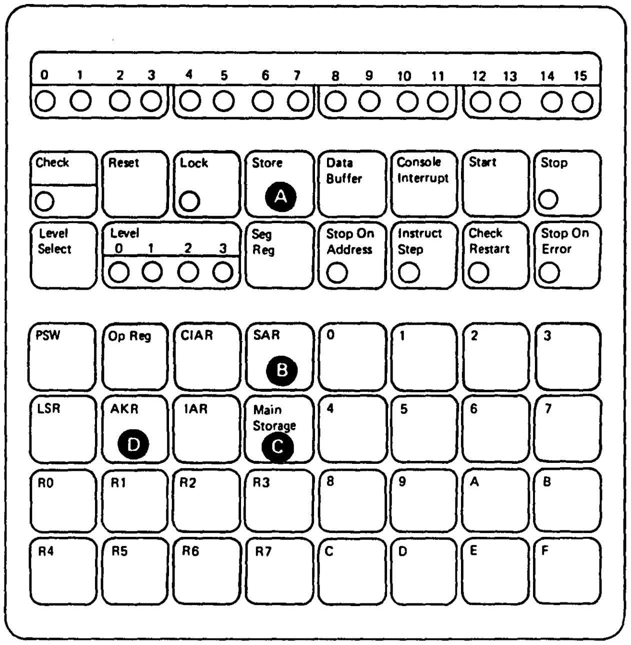
The processor must be in stop state.

If the storage address relocation translator is enabled, start at step 1; otherwise, start at step 5.

> **Note:** If steps 1 through 4 of the procedure are used, it is assumed that the operator has a thorough knowledge of the relocation translator and the storage mapping assigned by the program.

1. Press the SAR key Ⓑ. The contents of SAR are displayed in the data display indicators.

2. Press the AKR key Ⓓ. The contents of the SAR AKR are displayed in the data display indicators.



3. Key in one hex character (value of 0 through F, which is the new address key). This character is displayed in bits 12–15 of the data display indicators.

4. Press the Store key Ⓐ. The new address key is stored into the SAR AKR.

5. Press the SAR key Ⓑ. The contents of the SAR are displayed in the data display indicators.

6. Key in the selected address (four hex characters). This address is displayed in the data display indicators.

7. Press the Store key Ⓐ. The address that is displayed is stored into the SAR.

8. Press the Main Storage key Ⓒ. The contents of the addressed storage location are displayed in the data display indicators and SAR is incremented by 2. Each time the Main Storage key is pressed, the location addressed by SAR is displayed in the data display indicators and then SAR is incremented by 2.

Notes:

1. If an invalid storage address occurs:

   a. The program check is suppressed.

   b. PSW bit 1 is set to 1.

   c. The Check indicator is turned on.

   d. PSW bit 1 set does not cause a class interrupt to occur upon entering the run state unless the check indicator is not reset. The bit is only an indication, to the operator, of an error while displaying main storage.

2. If a storage location with bad storage parity occurs:

   a. The program check is suppressed.

   b. PSW bit 8 is set to 1.

   c. The Check indicator is turned on.

   d. PSW bit 8 set does not cause a class interrupt to occur upon entering the run state unless the check indicator is not reset. The bit is only an indication, to the operator, of an error while displaying main storage.

Storing Into Main Storage

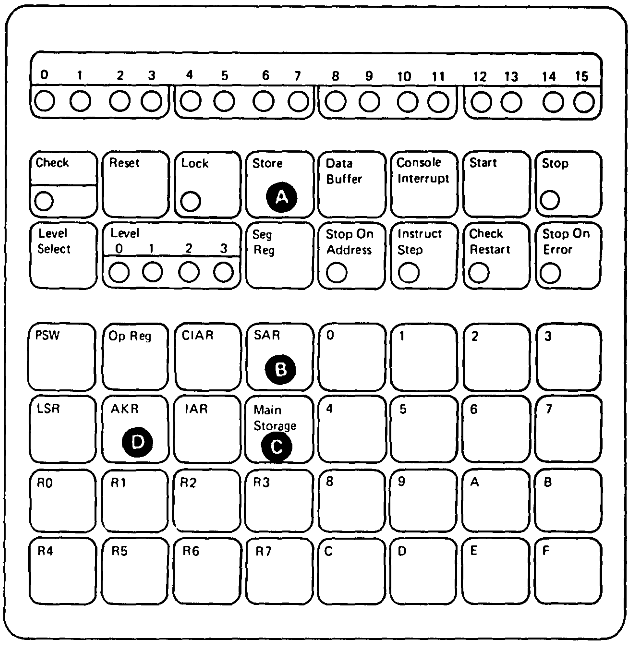
The processor must be in stop state.

If the storage address relocation translator is enabled, start at step 1; otherwise, start at step 4.

> **Note:** If steps 1 through 3 of the procedure are used, it is assumed that the operator has a thorough knowledge of the relocation translator and the storage mapping assigned by the program.

1. Press the SAR key Ⓑ. The contents of SAR are displayed in the data display indicators.

2. Press the AKR key Ⓓ. The contents of the SAR AKR are displayed in the data display indicators.



3. Key in one hex character (a value of 0–F which is the new address key). This character is displayed in bits 12–15 of the data display indicators.

4. Press the Store key Ⓐ. The new address key is stored into the SAR AKR.

5. Press the SAR key Ⓑ. The current contents of the SAR are displayed in the data display indicators.

6. Key in the selected address (four hex characters). The address is displayed in the data display indicators.

7. Press the Store key Ⓐ. The address displayed in the data display indicators is stored into the SAR.

8. Press the Main Storage key Ⓒ. The contents of the addressed storage location are displayed in the data display indicators.

9. Key in the data that is to be stored into main storage. This data is displayed in the data display indicators.

10. Press the Store key Ⓐ. The data that is displayed is stored at the selected storage location and SAR is incremented by 2. Repeat steps 9 and 10 to store in sequential storage word addresses, or repeat steps 8, 9, and 10 if sequential storage words are to be displayed before alteration.

Chapter 4. Diagnose (DIAG) Instruction

The DIAG instruction is used for controlling or testing various hardware functions.

DIAG	ubyte	
Op code	*Func*	*Parameter field*
0 1 1 0 0	1 0 1	
0 4	*5 7*	*8 15*

Additional words when accessing local storage

	Loc stor reg addr
0 0 0 0 0 0 0 0	
16 23	*24 31*

Immediate data field
32 47

The parameter field is used to define and select the functions of the DIAG instruction. The bits in the parameter field are as follows:

Parameter field

	0 0				
8 9	*10 11*	*12*	*13*	*14*	*15*

Note: Bits 10 and 11 must always be set to 0's.

Bits	**Value**	**Function**
8–9	00	Storage select (word)
	01	Storage select (byte/error correction code bits)
	10	Local storage register select
	11	Channel select
10	0	Not used (must be set to 0)
11	0	Not used (must be set to 0)
12	0	Storage-to-register data transfer
	1	Register-to-storage data transfer
13	0	Enable all other parameter bit functions
	1	Set system ID (all other parameter bit functions disabled)
14	0	Disable (Error correction code, error log, channel-interrupt requests, and channel cycle-steal requests)
	1	Enable (Error correction code, error log, channel-interrupt requests, and channel cycle-steal requests)
15	0	Enable all other parameter bit functions
	1	Error log select (storage select, local storage register select, and channel select functions disabled)

Storage Select

The storage select function provides for testing of the error correction code (ECC) generation, single error correction, and double error detection on the storage card.

> **Note:** During a write storage cycle, ECC generates six code bits for the 16-bit data word written (for byte writes, a read must first occur to form the 16-bit data word) thus creating a 22-bit word in storage. These code bits provide for the single error correction/double error detection capability on the read storage cycle. When a double error in storage is detected on a processor read, a machine check interrupt occurs with PSW bit 8 set to 1 (storage parity error).

Storage select can be either by word or by byte/ECC code bits. Parameter-field bits that are common to word or byte/ECC code bits selection are described in the following:

Bit 12: Specifies the direction of the data transfer.

- Bit 12=0. Transfer is from main storage to a register (read storage).
- Bit 12=1. Transfer is from a register to main storage (write storage).

Bit 13: Must be set to 0.

Bit 14: Specifies ECC generation, single error correction, and double error detection.

- Bit 14=0. ECC is not generated, single errors are not corrected, and double errors are not detected.
- Bit 14=1. ECC is generated, single errors are corrected, and double errors are detected.

Bit 15: Must be set to 0.

Storage Select Word

Parameter-field bits 8 and 9 are set to 0's.

The storage address for this data transfer cycle is contained in register 7 of the current priority level; the data is contained in register 0 of the current priority level. Two bytes of data are transferred.

Bit 14: Disable/enable.

- Bit 14=0. Disable.

  Read storage — Single errors are not corrected, and double errors are not detected.

  Write storage — ECC is not generated. Only the 16-bit data word is updated in storage; the six code bits in the 22-bit word remain unchanged.

- Bit 14=1. Enable.

  Read storage — Single errors are corrected, and double errors are detected.

  Write storage — ECC is generated.

When bit 14=1 (enable), a normal read or write word (bit 12 set to 0 or 1) occurs in storage.

Storage Select Byte/ECC Code Bits

Parameter-field bit 8 is set to 0 and bit 9 is set to 1.

The storage address for this data transfer cycle is contained in register 7 of the current priority level.

Bit 14: Disable/enable.

- Bit 14=0. Disable.

  Read storage — The six code bits are transferred from the 22-bit storage word to bits 10–15 of current priority level register 0. Bits 0–9 of register 0 are set to 0's. Single errors are not corrected, and double errors are not detected.

  Write storage — Bits 10–15 of current priority level register 0 are transferred to the six code bit positions of the word in storage. The 16 data bits in the 22-bit word remain unchanged. Bits 0–9 of register 0 are ignored.

- Bit 14=1. Enable.

  Read storage — The storage data byte is transferred from storage into bits 8–15 of current priority level register 0. Bits 0–7 of register 0 are set to 0's. Single errors are corrected, and double errors are detected.

  Write storage — The data byte in bits 8–15 of current priority level register 0 is transferred to storage. Bits 0–7 of register 0 are ignored. ECC is generated.

When bit 14=1 (enable), a normal read or write byte (bit 12 set to 0 or 1) occurs in storage.

Local Storage Register Select

The local storage register select function permits the transfer of data between main storage and any local storage register. The recommended use of this function is to read the error log. (See Appendix C.) To select this function, parameter-field bit 8 is set to 1 and bit 9 is set to 0.

> **Note:** The processor uses the local storage registers to store various machine parameters. Changing these parameters is not recommended because the results cannot be predicted.

The DIAG instruction has two additional words appended when this function is specified.

Additional words when accessing local storage

0 0 0 0 0 0 0 0	*Loc stor reg addr*
16 23	*24 31*

Immediate data field
32 47

The bits in the two additional words are defined as follows:

Bits	**Significance**
16–23	Not used (must be set to 0's)
24–31	Local storage register address (00–FF hex)
32–47	Immediate data to be transferred

Parameter-field bits that are used with this function are defined as follows:

Bit 12: Specifies the direction of the data transfer.

- Bit 12=0. Transfer is from the immediate data field to the specified local storage register.

- Bit 12=1. Transfer is from the specified local storage register to the immediate data field.

Bit 13: Must be set to 0.

Bit 15: Must be set to 0.

Channel Select

The channel select function is determined by parameter field bits 8 and 9 being set to 1's.

This function, with bit 14 of the parameter field set to 0 (disable), inhibits and logically isolates I/O interrupts and cycle-steal operations. With bit 14 of the parameter field set to 1 (enable), this function allows I/O interrupts under the control of the summary mask, and cycle-steal operations are enabled.

Parameter-field bit functions are defined as follows:

Bit 13: Must be set to 0.

Bit 14: Specifies whether channel priority interrupts and cycle-steal requests are disabled or enabled.

- Bit 14=0. Channel priority interrupts and cycle-steal requests are disabled.

- Bit 14=1. Channel priority interrupts and cycle-steal requests are enabled.

Bit 15: Must be set to 0.

> **Note:** Pressing the Start button while in the stop state or executing any instruction that causes a level status block to be loaded (LEX instruction, SELB instruction, class interrupt, etc.) returns priority interrupt masking to program control. Also, the following operations cause cycle-stealing to be resumed:
>
> - Setting or resetting Stop-on-Address mode.
>
> - Performing the EN, DIS, SESR, or CPSR instruction.

Set System ID

The set system ID function is selected by parameter field bit 13 being set to 1.

This function sets the system ID into register 0 of the current priority level. The system ID for the processor is hex 0206.

Register 0 is set as follows:

0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0
0															*15*

> **Note:** When this function is selected, all other parameter-field functions are ignored.

Error Log Select

The error log select function provides a control to prevent logging of errors generated by diagnostic tests. To select this function, parameter-field bit 15 must be set to 1.

Parameter-field bit functions are defined as follows:

Bit 13: Must be set to 0.

Bit 14: Enables or disables error logging.

- Bit 14=0. Error logging is disabled.

- Bit 14=1. Error logging is enabled.

Bit 15: Must be set to 1.

> **Note:** If error log select is specified, storage select, local storage register select, and channel select functions are disabled.

Indicators

Indicators are not changed by the DIAG instruction; however, the data in local storage registers may be changed. Refer to "Local Storage Register Select" in this section.

Program-Check Condition

The DIAG instruction is a privileged instruction. If this instruction is encountered during the problem state, the instruction is suppressed, a program-check interrupt occurs, and the privilege violate bit (bit 2) in the PSW is set to 1.

Appendix A. Instruction Execution Times

The 4956 Processor Models E and E10 execute instructions approximately 50% faster than previous models of the 4956.

Several factors have an affect on instruction throughput. Some of these factors are:

- Instruction mix
- Storage refresh
- Channel load interference
- Wait state
- Timer housekeeping
- Synchronization of the channel interface to the storage interface
- Correction of a single-bit error in storage.

Execution times for individual instructions can be calculated from the following information.

Figure A-1 shows the additional time required when executing register/storage instructions or storage/storage instructions and assembler syntax for address mode. The symbols used in Figure A-1 and in the remainder of this appendix are defined as follows:

Symbol	Meaning
N	Number of bytes moved, filled, scanned, or compared
NR	Number of registers set or copied
NS	Number of shifts
RS	Additional addressing-mode time for register/storage instructions
SS	Additional addressing-mode time for storage/storage instructions
*	Indirect address

- RS—the additional time for register/storage instructions

AM	RB	Time (microseconds)
00	Note 1	0.00
01	Note 1	0.00
10	Note 1	0.35
11	RB = 0	0.35
	RB ≠ 0	1.30

- SS—the additional time for storage/storage instructions (all combinations of AM1 and AM2 are shown below)

AM1	RB1	AM2	RB2	Time (microseconds)
00	Note 1	00	Note 1	0.00
00	Note 1	01	Note 1	0.00
00	Note 1	10	Note 1	0.00
00	Note 1	11	= 0	0.70
			≠ 0	1.30
01	Note 1	00	Note 1	0.00
01	Note 1	01	Note 1	0.00
01	Note 1	10	Note 1	0.40
01	Note 1	11	= 0	0.70
			≠ 0	1.30
10	Note 1	00	Note 1	0.00
10	Note 1	01	Note 1	0.00
10	Note 1	10	Note 1	0.40
10	Note 1	11	= 0	0.95
			≠ 0	1.45
11	=0	00	Note 1	0.55
11	=0	01	Note 1	1.00
11	=0	10	Note 1	1.00
11	=0	11	= 0	1.95
			≠ 0	2.30
11	≠0	00	Note 1	1.30
11	≠0	01	Note 1	1.75
11	≠0	10	Note 1	1.75
11	≠0	11	= 0	2.30
			≠ 0	3.05

- Assembler syntax for address modes (Note 2)

Assembler syntax		Address modes	
addr4	*addr5*	*AM, AM1, or AM2*	
(reg$^{0-3}$)	(reg)	00	Note 1
(reg$^{0-3}$)+	(reg)+	01	Note 1
addr	addr	10	Note 1
(reg$^{1-3}$,waddr)	(reg$^{1-7}$,waddr)	10	Note 1
addr*	addr*	11	RB=0
disp1(reg$^{1-3}$,disp2)*	disp1(reg$^{1-7}$,disp2)*	11	RB≠0
disp(reg$^{1-3}$)*	disp(reg$^{1-7}$)	11	RB≠0
(reg$^{1-3}$)*	(reg$^{1-7}$)*	11	RB≠0
(reg$^{1-3}$,disp)*	(reg$^{1-7}$,disp)*	11	RB≠0

Notes:

1. The value of RB does not affect the timings.
2. Register/storage instructions use assembler syntax *addr4* for address mode (AM).
   Storage/storage instructions use assembler syntax:
   - *addr5* for address mode for operand 1 (AM1)
   - *addr4* for address mode for operand 2 (AM2)

Figure A-1. Additional Instruction Times and Assembler Syntax for Address Mode

The following notes and tables are used to determine instruction execution times:

Notes:

1. If a shift of 32 or greater is specified in Reg., Reg. format:

   SLCD = 6.40
   SLLD = 4.50
   SRLD = 2.70
   SRAD = 3.15

2. If the opcode is SLT or SLTD and the shifted register is initially equal to zero, do not add the 0.45ns or 0.9 factor.

3. L is equal to the number of levels examined before an in process bit is found ON or wait state is reached.

4. Diagnose

1	STG Write	4.35 (Word)
2	STG Read	4.40 (Word)
3	Test Register Stack	
	(A) STG to STK	4.20
	(B) STK to STG	4.05
4	Enable I/O Interface	3.15
5	Disable I/O Interface	3.15
6	Set System ID	1.80
7	Enable/Disable Error Log	2.25

5. Set Level Block

IP = In-Process Bit   SL=Set Level   SC=Current Level

IP OFF	SL<CL	11.70+AM
IP OFF	SL=CL	20.05+AM (see note)
IP OFF	SL>CL	11.55+AM
IP ON	SL<CL	11.70+AM
IP ON	SL=CL	14.65+AM
IP ON	SL>CL	14.65+AM

Note: Worst case, assumes lexing from level–0 to level–3

I/O Interrupt

14.7 micro-seconds from wait state (Request to first instruction)

AM times =

AM11	RB=0	0.65
AM11	RB≠0	1.45
AM00	RB=X	0.0
AM10	RB=X	0.0
AM01	RB=X	0.0

Note: X indicates don't care

6. If running in address stop mode $= 4.15 + N(1.75) + RS$

7. Bits 0–3 of the IDCB are decoded to determine Halt, Read, or Write type operations to the channel or device. The times reflect typical delays induced by the attachment and/or device to which the command is directed but exclude any channel repowering. Times given are for Read ID; add 1.65 microseconds for Write.

8. Load Multiple and Branch (R7 only $4.95+RS$)

   For each specified register from 0–6, add an additional 0.65 per register.

9. Store Multiple (R7 only $4.95+RS$)

   For each specified register from 1–6, add an additional 0.65 per register. When register zero only is specified, add 0.50.

10. For all the following VFL opcodes, $N =$ the number of bytes moved, filled, scanned or compared.

    MVFN has three cases:

    Case 1: Both addresses (in R1, R2) are even or both addresses are odd and $N>3$. If N is odd, round it up to the next higher even number. $5.00 + 1.05\ N/2$

    Case 2: If $N<4$ or if one address is odd and the other address is even ($N>0$). $2.55 + 1.30N$

    Case 3: If $N=0$ -1.40

    MVFD has four cases:

    Case 1 Both addresses (in R1 and R2) are even and $N>3$. If N is odd, round it up to the next higher even number. $5.10 + 1.05\ N/2$

    Case 2 Both addresses (in R1 and R2) are odd and $N>3$. If N is odd, round it up to the next higher even number. $5.55 + 1.05\ N/2$

    Case 3 If $0<N<4$ or if one address is odd and the other is even. $2.55 + 1.30N$

    Case 4 If $N=0$ 1.40

    Note: N is equal to the number of bytes moved.

11. Minus 1.35 if $N=0$. Minus 0.55 if terminated before $R7=0$.

12. Minus 1.35 if $N=0$.

13. Minus 1.40 if $N=0$. Minus 0.45 if terminated with $R7>1$.

The following tables show the instructions in alphabetical sequence based on assembler mnemonic.

Instruction execution times			
Mnemonic	**Instruction name**	**Syntax**	**Execution time (microseconds)**
AA	Add Address	raddr,reg[,reg]	0.90
		raddr,addr4	1.95+RS
AB	Add Byte	reg,addr4	1.40+RS
		addr4,reg	1.45+RS
ABI	Add Byte Immediate	byte,reg	0.45
ACY	Add Carry Register	reg	0.45
AD	Add Doubleword	reg,addr4	2.75+RS
		addr4,reg	1.85+RS
		addr5,addr4	4.35+SS
ARIB ARIBOFF * ARIBON			
AW	Add Word	reg,reg	0.50
		reg,addr4	1.55+RS
		addr4,reg	1.10+RS
		longaddr,reg	1.10
		longaddr*,reg	2.15
		addr5,addr4	2.45+SS
AWCY	Add Word with Carry	reg,reg	0.50
AWI	Add Word Immediate	word,reg[,reg]	0.90
		word,addr4	1.95+RS
B	Branch Unconditional	longaddr	1.40
		longaddr*	2.45
BAL	Branch and Link	longaddr,reg	1.75
		longaddr*,reg	2.00
BALS	Branch and Link Short	(reg,jdisp)*	2.00
		(reg)*	2.00
		addr*	2.00
BALX	Branch and Link External	longaddr, reg	1.75
		longaddr*, reg	2.00
BC	Branch on Condition	cond, longaddr	0.90 No branch 1.40 Branch taken
		cond,longaddr*	0.90 No branch 2.45 Branch taken
BCC	Branch on Condition Code	cond,longaddr	0.90 No branch 2.45 Branch taken
		cond,longaddr*	0.90 No branch 2.45 Branch taken
BCY	Branch on Carry	Same as for BC	Same time as for BC
BE	Branch on Equal		
BER	Branch on Error		
BEV	Branch on Even		
BGE	Branch on Arithmetically Greater Than or Equal		

*The ARIB instructions are in a separate table that follows this set of tables.

Instruction execution times			
Mnemonic	Instruction name	Syntax	Execution time (microseconds)
BGT	Branch on Arithmetically Greater Than		Same time as for BC
BLE	Branch on Arithmetically Less Than or Equal		
BLGE	Branch on Logically Greater Than or Equal		
BLGT	Branch on Logically Greater Than		
BLLE	Branch on Logically Less Than or Equal		
BLLT	Branch on Logically Less Than		
BLT	Branch on Arithmetically Less Than		
BMIX	Branch if Mixed		
BN	Branch on Negative		
BNC	Branch on Not Condition	cond,longaddr cond,longaddr*	0.90 No branch 1.40 Branch taken 0.90 No branch 2.45 Branch taken
BNCC	Branch on Not Condition Code	cond,longaddr cond,longaddr*	0.90 No branch 1.40 Branch taken 0.90 No branch 2.45 Branch taken
BNCY	Branch on No Carry		Same time as for BNC
BNE	Branch on Not Equal		
BNER	Branch if Not Error		
BNEV	Branch on Not Even		
BNMIX	Branch if Not Mixed		
BNN	Branch on Not Negative		
BNOFF	Branch if Not Off		
BNON	Branch of Not On	longaddr longaddr*	0.90 No branch 1.40 Branch taken 0.90 No branch 2.45 Branch taken
BNOV	Branch on Not Overflow		
BNP	Branch on Not Positive		
BNZ	Branch on Not Zero		Same time as BNC
BOFF	Branch if Off		
BON	Branch if On	longaddr longaddr*	0.90 No branch 1.40 Branch taken 0.90 No branch 2.45 Branch taken
BOV	Branch on Overflow		
BP	Branch on Positive		
BX	Branch External	vcon	Same time as for B
BXS	Branch Indexed Short	(reg^1 —^{7},jdisp) (reg^1 —^{7}) addr	1.40 1.40 1.40
BZ	Branch on Zero		Same time as for BC

Instruction execution times			
Mnemonic	Instruction name	Syntax	Execution time (microseconds)
CA	Compare Address	raddr,reg	0.90
		raddr,addr4	1.50+RS
CB	Compare Byte	addr4,reg	1.35+RS
		addr5,addr4	2.15+SS
CBI	Compare Byte Immediate	byte,reg	0.45
CD	Compare Doubleword	addr4,reg	2.15+RS
		addr5,addr4	4.20+SS
CFED	Compare Byte Field Equal and Decrement	(reg),(reg)	2.40+1.75N Note 11
CFEN	Compare Byte Field Equal and Increment	(reg),(reg)	2.40+1.75N Note 11
CFNED	Compare Byte Field Not Equal and Decrement	(reg),(reg)	2.40+1.75N Note 11
CFNEN	Compare Byte Field Not Equal and Increment	(reg),(reg)	2.40+1.75N Note 11
CMR	Complement Register	reg[,reg]	0.90
CPAKR	Copy Address Key Register	addr4	1.35+RS
		reg	0.90
CPCL	Copy Current Level	reg	0.90
CPCLK	Copy Clock	reg	0.90
CPCMP	Copy Comparator	reg	0.90
CPCON	Copy Console Data Buffer	reg	0.45
CPIMR	Copy Interrupt Mask Register	addr4	1.35+RS
CPIPF	Copy In-Process Flags	addr4	1.35+RS
CPISK	Copy Instruction Space Key	addr4	1.35+RS
		reg	0.90
CPLB	Copy Level Block	reg,addr4	10.80+RS
CPLSR	Copy Level Status Register	reg	0.45
CPOOK	Copy Operand 1 Key	addr4	1.35+RS
		reg	0.90
CPOTK	Copy Operand 2 Key	addr4	1.35+RS
		reg	0.90
CPPSR	Copy Processor Status and Reset	addr4	2.30+RS
CPSK	Copy Storage Key (no op)	reg,addr4	0.90+RS
CPSR	Copy Segmentation Register	reg,addr4	2.25+NR(2.35)+RS
CW	Compare Word	reg,reg	0.50
		addr4,reg	1.20+RS
		addr5,addr4	2.45+SS
CWI	Compare Word Immediate	word,reg	0.90
		word,addr4	1.50+RS

Instruction execution times

Mnemonic	Instruction name	Syntax	Execution time (microseconds)
DB	Divide Byte	addr4,reg	3.85+RS Minimum 33.55+RS Maximum
DD	Divide Doubleword	addr4,reg	4.15+RS Minimum 56.45+RS Maximum
DIAG	Diagnose	ubyte	Note 4
DIS	Disable	ubyte	Op bits 10 12 13 14 15 Times 1 0 0 0 0 2.70 0 0 0 0 1 2.70 0 0 0 1 0 2.70 0 0 1 0 0 2.70 0 1 0 0 0 2.25
DW	Divide Word	addr4,reg	3.40+RS Minimum 33.10+RS Maximum
EN	Enable	ubyte	Op bits 10 12 13 14 15 Times 1 0 0 0 0 2.70 0 0 0 0 1 2.70 0 0 0 1 0 2.70 0 0 1 0 0 2.70 0 1 0 0 0 2.70

Instruction execution times

Mnemonic	Instruction name	Syntax	Execution time (microseconds)
FFD	Fill Byte Field and Decrement	reg,(reg)	2.75+0.9N Note 12
FFN	Fill Byte Field and Increment	reg,(reg)	2.75+0.9N Note 12
IO	Operate I/O	longaddr	5.60 Note 7
		longaddr*	6.15 Note 7
IOPK	Interchange Operand Keys Keys		1.35
IR	Interchange Registers	reg,reg	0.95
J	Jump Unconditional	jdisp	1.40
		jaddr	1.40
JAL	Jump and Link	jdisp,reg	1.40
		jaddr,reg	1.40
JC	Jump on Condition	cond,jdisp	0.45
		cond,jaddr	0.45
		jump not taken	
		cond,jaddr	1.40
		cond,jdisp	1.40
		jump taken	
JCT	Jump on Count	jdisp,reg	1.80 Minimum 2.15 Maximum
		jaddr,reg	1.80 Minimum 2.15 Maximum
JCY	Jump on Carry	jdisp,jaddr	Same time as JC
JE	Jump on Equal	jdisp,jaddr	
JEV	Jump on Even	jdisp,jaddr	
JGE	Jump on Arithmetically	jdisp,jaddr	
	Greater Than or Equal	jdisp,jaddr	
JGT	Jump on Arithmetically	jdisp,jaddr	
	Greater Than	jdisp,jaddr	
JLE	Jump on Arithmetically	jdisp,jaddr	
	Less Than or Equal	jdisp,jaddr	
JLGE	Jump on Logically	jdisp,jaddr	
	Greater Than or Equal	jdisp,jaddr	
JLGT	Jump on Logically	jdisp,jaddr	
	Greater Than	jdisp,jaddr	
JLLE	Jump on Logically	jdisp,jaddr	
	Less Than or Equal	jdisp,jaddr	
JLLT	Jump on Logically	jdisp,jaddr	
	Less Than	jdisp,jaddr	

Instruction execution times

Mnemonic	Instruction name	Syntax	Execution time (microseconds)	
JLT	Jump on Arithmetically	jdisp,jaddr	Same time as JC	
	Less Than	jdisp,jaddr		
JMIX	Jump if Mixed	jdisp,jaddr		
JN	Jump on Negative	jdisp,jaddr		
JNC	Jump on Not Condition	cond,jdisp		
		jump not taken	0.45	
		cond,jaddr		
		jump taken	1.40	
JNCY	Jump on No Carry		Same time as JNC	
JNE	Jump on Not Equal			
JNEV	Jump on Not Even			
JNMIX	Jump if Not Mixed			
JNN	Jump on Not Negative			
JNOFF	Jump if Not Off			
JNON	Jump if Not On			
JNP	Jump on Not Positive			
JNZ	Jump no Not Zero			
JOFF	Jump if Off			
JON	Jump if On			
JP	Jump on Positive			
JZ	Jump on Zero			
LEX	Level Exit	[ubyte]	6.75+1.30L	Note 3
LMB	Load Multiple and Branch	addr4	4.95+RS	Note 3
MB	Multiply Byte	addr4,reg	4.25+RS	Minimum
			8.75+RS	Maximum
MD	Multiply Doubleword	addr4,reg	6.30+RS	Minimum
			28.25+RS	Maximum
MVA	Move Address	addr4,reg	0.90	
		raddr,addr4	1.35+RS	
MVB	Move Byte	reg,addr4	1.35+RS	
		addr4,reg	1.05+RS	
		addr5,addr4	1.65+SS	
MVBI	Move Byte Immediate	byte,reg	0.45	
MVBZ	Move Byte and Zero	addr4,reg	1.50+RS	
MVD	Move Doubleword	reg,addr4	1.80+RS	
		addr4,reg	2.60+RS	
		addr5,addr4	3.30+SS	
MVDZ	Move Doubleword and Zero	addr4,reg	2.80+RS	
MVFD	Move Byte Field and Decrement	(reg),(reg)	Note 10	
MVFN	Move Byte Field and Increment	(reg),(reg)	Note 10	

Instruction execution times			
Mnemonic	Instruction name	Syntax	Execution time (microseconds)
MVW	Move Word	reg,reg	0.45
		reg,addr4	1.35+RS
		addr4,reg	1.05+RS
		addr5,addr4	2.10+SS
		reg,longaddr	1.35
		reg,longaddr*	2.35
		longaddr,reg	1.05
		longaddr*,reg	2.10
MVWI	Move Word Immediate	word,reg	0.90
		word,addr4	1.35+RS
MVWS	Move Word Short	reg,shortaddr	0.90
		reg,shortaddr*	1.60
		shortaddr,reg	1.00
		shortaddr*,reg	1.55
MVWZ	Move Word and Zero	addr4,reg	1.50+RS
MW	Multiply Word	addr4,reg	3.80+RS Minimum
			14.65+RS Maximum
NOP	No Operation		1.40
NWI	AND Word Immediate	word,reg[,reg]	0.95
OB	OR Byte	reg,addr4	1.40+RS
		addr4,reg	1.55+RS
		addr5,addr4	2.45+SS
OD	OR Doubleword	reg,addr4	1.85+RS
		addr4,reg	2.75+RS
		addr5,addr4	4.40+SS
OW	OR Word	reg,reg	0.50
		reg,addr4	1.55+RS
		addr4,reg	1.10+RS
		longaddr,reg	1.10
		longaddr*,reg	2.15
		addr5,addr4	2.45+SS
OWI	OR Word Immediate	word,reg[,reg]	0.90
		word,addr4	1.95+RS
PB	Pop Byte	addr4,reg	3.70+RS
PD	Pop Doubleword	addr4,reg	4.75+RS
PSB	Push Byte	reg,addr4	4.30+RS
PSD	Push Doubleword	reg,addr4	4.65+RS
PSW	Push Word	reg,addr4	3.60+RS
PW	Pop Word	addr4,reg	3.25+RS
RBTB	Reset Bits Byte	reg,addr4	1.40+RS
		addr4,reg	1.55+RS
		addr5,addr4	2.45+SS
RBTD	Reset Bits Doubleword	reg,addr4	1.85+RS
		addr4,reg	2.75+RS
		addr5,addr4	4.40+SS

Instruction execution times

Mnemonic	Instruction name	Syntax	Execution time (microseconds)
RBTW	Reset Bits Word	reg,reg	0.50
		reg,addr4	1.10+RS
		addr4,reg	1.15+RS
		longaddr,reg	1.10
		longaddr*,reg	2.15
		addr5,addr4	2.45+SS
RBTWI	Reset Bits Word Immediate	word,reg|,reg|	0.90
		word,addr4	1.95+RS
SA	Subtract Address	raddr,reg|,reg|	0.90
		raddr,addr4	1.95+RS
SB	Subtract Byte	reg,addr4	1.40+RS
		addr4,reg	1.60+RS
SBTB	Set Bits Byte	reg,addr4	1.40+RS
		addr4,reg	1.55+RS
		addr5,addr4	2.45+RS
SBTD	Set Bits Doubleword	reg,addr4	1.85+RS
		addr4,reg	2.75+RS
		addr5,addr4	4.40+SS
SBTW	Set Bits Word	reg,reg	0.50
		reg,addr4	1.55+RS
		addr4,reg	1.10+RS
		longaddr,reg	1.10
		longaddr*,reg	2.15
		addr5,addr4	2.45+SS
SBTWI	Set Bits Word Immediate	word,reg|,reg|	0.90
		word,addr4	1.95+RS
SCY	Subtract Carry Indicator	reg	0.45
SD	Subtract Doubleword	reg,addr4	2.75+RS
		addr4,reg	1.85+RS
		addr5,addr4	4.35+SS
SEAKR	Set Address Key Register	addr4	2.25+RS
		reg	2.25
SECLK	Set Clock	reg	0.90
SECMP	Set Comparator	reg	1.35
SECON	Set Console Data Lights	reg	2.35
SEIMR	Set Interrupt Mask Register	addr4	1.80+RS
SEIND	Set Indicators	reg	0.90
SEISK	Set Instruction Space Key	addr4	2.25+RS
		reg	2.25
SELB	Set Level Block	reg,addr4	Note 5
SEOOK	Set Operand 1 Key	addr4	2.25+RS
		reg	2.25

Instruction execution times

Mnemonic	Instruction name	Syntax	Execution time (microseconds)
SEOTK	Set Operand 2 Key	addr4	2.25+RS
		reg	2.25
SESK	Set Storage Key	reg,addr4	0.90+RS
SESR	Set Segmentation Register	reg,addr4	2.40+NR(1.75)+RS Note 6
SFED	Scan Byte Field Equal and Decrement	reg,(reg)	2.80+0.95N Note 13
SFEN	Scan Byte Field Equal and Increment	reg,(reg)	2.80+0.95N Note 13
SFNED	Scan Byte Field Not Equal and Decrement	reg,(reg)	2.80+0.95N Note 13
SFNEN	Scan Byte Field Not Equal and Increment	reg,(reg)	2.80+0.95N Note 13
SLC	Shift Left Circular	cnt16,reg	3.15
		reg,reg	2.70
SLCD	Shift Left Circular Double	cnt31,reg	6.40 Note 1
		reg,reg	5.95 Note 1
SLL	Shift Left Logical	cnt16,reg	3.60 Minimum
			4.05 Maximum
		reg,reg	3.15 Minimum
			3.60 Maximum
SLLD	Shift Left Logical Double	cnt31,reg	7.25 Minimum Note 1
			8.65 Maximum Note 1
		reg,reg	6.80 Minimum Note 1
			8.20 Maximum Note 1
SLT	Shift Left and Test	reg,reg	2.30+0.45NS Note 2
SLTD	Shift Left and Test Double	reg,reg	2.70+0.09NS Note 2
SRA	Shift Right Arithmetic	cnt16,reg	3.15
		reg,reg	2.70
SRAD	Shift Right Arithmetic Double	cnt31,reg	6.35 Minimum Note 1
			6.40 Maximum Note 1
		reg,reg	5.90 Minimum Note 1
			5.95 Maximum Note 1
SRL	Shift Right Logical	cnt16,reg	3.15
		reg,reg	2.70
SRLD	Shift Right Logical Double	cnt31,reg	5.90 Minimum Note 1
			6.30 Maximum Note 1
		reg,reg	5.45 Minimum Note 1
			5.85 Maximum Note 1
STM	Store Multiple	reg,addr4[,abcnt]	4.95+RS Note 9
STOP	Stop	[ubyte]	1.35
SVC	Supervisor Call	ubyte	11.95

Instruction execution times			
Mnemonic	Instruction name	Syntax	Execution time (microseconds)
SW	Subtract Word	reg,reg	0.50
		reg,addr4	1.55+RS
		addr4,reg	1.20+RS
		longaddr,reg	1.10
		longaddr*,reg	2.15
		addr5,addr4	2.45+SS
SWCY	Subtract Word with Carry	reg,reg	0.55
SWI	Subtract Word Immediate	word,reg[,reg]	1.95+RS
		word,addr4	0.90
TBT	Test Bit	(reg,bitdisp)	1.20
TBTR	Test Bit and Reset	(reg,bitdisp)	2.10
TBTS	Test Bit and Set	(reg,bitdisp)	2.10
TBTV	Test Bit and Invert	(reg,bitdisp)	2.10
TWI	Test Word Immediate	word,reg	1.35
		word,addr4	2.10+RS
VR	Invert Register	reg[,reg]	0.45
XB	Exclusive OR Byte	reg,addr4	1.40+RS
		addr4,reg	1.55+RS
XD	Exclusive OR Doubleword	reg,addr4	2.75+RS
		addr4,reg	1.85+RS
XW	Exclusive OR Word	reg,reg	0.50
		reg,addr4	1.55+RS
		addr4,reg	1.10+RS
		longaddr,reg	1.10
		longaddr*,reg	2.15
XWI	Exclusive OR Word Immediate	word,reg[,reg]	0.90

ARIB, ARIBON, and ARIBOFF instruction execution times			
Mnemonic	Instruction name	Syntax	Execution time (microseconds)
ARIB	Address Resolution with Indirect Branch	disp1,disp2, Tbladdr	
		addr,addr	6.15
		index,addr	7.40
		addr,index	7.40
		index,index	9.00
		addr,immed	6.15
		index,immed	7.85
ARIBON	Address Resolution with Indirect Branch On		0.90
ARIBOFF	Address Resolution with Indirect Branch Off		0.90

Floating-point instruction execution times

Mnemonic	Instruction name	Syntax	Execution time (microseconds)	
CPFLB	Copy Floating Level Block	freg,addr4	15.75+RS	
FA	Floating Add	addr4,freg	7.25+RS	Minimum
			8.20+RS	Maximum
		freg,freg	6.80	Minimum
			7.75	Maximum
FAD	Floating Add Double	addr4,freg	9.20+RS	Minimum
			10.20+RS	Maximum
		freg,freg	8.35	Minimum
			9.35	Maximum
FC	Floating Compare	freg,freg	8.20	Minimum
			9.10	Maximum
FCD	Floating Compare Double	freg,freg	9.10	Minimum
			11.80	Maximum
FD	Floating Divide	addr4,freg	6.35+RS	Minimum
			54.00+RS	Maximum
		freg,freg	5.90	Minimum
			53.55	Maximum
FDD	Floating Divide Double	addr4,freg	7.35+RS	Minimum
			162.70+RS	Maximum
		freg,freg	6.50	Minimum
			161.85	Maximum
FM	Floating Multiply	addr4,freg	4.95+RS	Minimum
			36.55+RS	Maximum
		freg,freg	5.90	Minimum
			36.10	Maximum
FMD	Floating Multiply Double	addr4,freg	6.30+RS	Minimum
			51.10+RS	Maximum
		freg,freg	6.50	Minimum
			49.80	Maximum
FMV	Floating Move	addr4,freg	2.25+RS	
		freg,freg	1.80	
		freg,addr4	1.90+RS	
FMVC	Floating Move and Convert	addr4,freg	3.60+RS	Minimum
			5.90+RS	Maximum
		freg,addr4	4.55+RS	Minimum
			5.85+RS	Maximum
FMVCD	Floating Move and Convert Double	addr4,freg	4.20+RS	Minimum
			7.00+RS	Maximum
		freg,addr4	5.65+RS	Minimum
			7.35+RS	Maximum
FMVD	Floating Move Double	addr4,freg	3.25+RS	
		freg,freg	2.35	
		freg,addr4	3.65+RS	

Floating-point instruction execution times

	Instruction		Execution	
Mnemonic	name	Syntax	(microseconds)	
FS	Floating Subtract	addr4,freg	7.70+RS	Minimum
			8.65+RS	Maximum
		freg,freg	7.25	Minimum
			8.20	Maximum
FSD	Floating Subtract Double	addr4,freg	9.20+RS	Minimum
			10.20+RS	Maximum
		freg,freg	8.35	Minimum
			9.35	Maximum
SEFLB	Set Floating Level Block	addr4,freg	18.80+RS	

Appendix B. Software Notes

Note 1

Instruction streams that are self-modifying cannot be guaranteed. An executing instruction cannot modify the next sequential three instruction stream words.

Example:

```
          R1=0000
          R2=LOC1
          R7=6

          FFN       R1,(R2)
LOC1      MVBI      2,R3
LOC2      SECON     R3
LOC3      LEX
```

This example illustrates a self-modifying instruction stream. The Fill Field and Increment (FFN) instruction moves 0000 to LOC1 through LOC3. A 0000 in machine code is a Add Byte Zero to Register Zero (ABI). The program is attempting to execute the ABI instruction instead of the sequence MVBI, SECON and LEX. This type of programming is not permitted.

Note 2

Four priority interrupt levels (0–3) are implemented in the processor. A Prepare command to levels 4–15 is executed so that condition code reporting occurs; however, the Prepare command is not executed at the addressed device and effectively results in a no-operation.

Note 3

There is no storage-protect feature in the 4956 processor. Execution of the Set Storage Key (SESK) and Copy Storage Key (CPSK) instructions results in a no-operation.

Note 4

Byte write operations to storage locations that contain two bit errors are not executed by the storage card. The storage location must first be corrected with a word write before the byte write operation can be executed correctly.

Note: After a successful power-on reset, all storage locations are initialized to some value.

Appendix C. Error Log

Purpose

The error log provides a history of errors that have occurred since power-on. The log is useful in isolating the cause of a particular problem. At power-on, all 64 entries in the error log are cleared; at any other time, the error log contains the most current 64 errors of the types of errors that are logged. This information is readily available to the CSR via the programmer's console or the Diagnose instruction.

Structure

Error log entries are placed in 64 local store registers (addresses hex 40 through 7F). The following errors are included in the log:

- Processor ISA check
- Specification check
- Storage parity error (double error detected in storage)
- I/O check with sequence indicator
- CPU control check
- Timer overrun (refer to *Stall Detector/Timer Overrun Error* later in this appendix)
- Storage protect check.

When one of these errors occurs, the SDR contents (the last word read or written by the processor to main storage) are loaded into local store address hex 05. Also included in the log are Operate I/O condition codes (busy after reset, command reject, and interface data check), and priority interrupt condition codes (exception, and attention and exception). To access the log, the lock key and a special code are used to display the error entries in the data lights. The Diagnose instruction may be used to dump the error log or local store address hex 05 to main storage for viewing.

Entries are placed in the error log starting with the first at local store address hex 40. Subsequent entries are placed in the log by incrementing the last previous address by 1. Local store address hex 0F contains the address of the last entry in the error log, in the low byte bits 8–15. (The high byte bits 0–7 contain machine parameters and must not be modified.) When all 64 entries have been written, the log wraps; that is, the next entry replaces the first. During power on, the local store address hex 0F is set to hex 003F and the error log is initialized to 0's. Thus, a location that contains hex 0000 indicates the end of the log, unless it is the first entry of a two-location log entry.

All errors are grouped into two types: machine check and program check. Timer overrun, CPU control check, I/O check, and storage parity error are machine check errors; processor ISA check, specification check, and storage protect check are considered program errors. Multiple errors within machine check and program check types are recorded. However, if any machine check condition occurs, program check errors are ignored.

Machine Check

Multiple machine check errors are logged in the following order:

1. Timer overrun (refer to *Stall Detector/Timer Overrun Error*)

2. CPU control check

3. I/O check with sequence indicator

4. Storage parity error.

If an I/O check occurs and the sequence indicator is set, no device address is logged (bits 8–15 of the log entry are set to 0's). If the sequence indicator is not set, the device address is placed in bits 8–15 of the log entry.

Program Check

Program check (processor ISA check, specification check, and storage protect check) generates two log entries for each error. The first entry to be logged is the contents of the CIAR. The second entry identifies the type of error, the supervisor state, and the last active address key. For all errors with two log entries, the second entry begins with binary 11. When displayed from the console, the second entry precedes the first, since the order of the display is from the last entry written.

Stall Detector/Timer Overrun Error

The stall detector is a unique and independent hardware timing mechanism in the 4956 processor. Its purpose is to check data integrity of the clock registers (correct time). If these registers are not updated every millisecond, the stall detector is activated. In addition, certain erroneous microcode branches that result in excessively high instruction execution time and impaired data integrity are detected by the stall detector.

When the stall detector activates, a machine check interrupt occurs with bit 10 (CPU control check) in the PSW set to 1. A timer overrun condition is then entered in the error log.

Format of Log Entries

Machine Check

Timer overrun:	0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0	(hex 1000)
CPU control check:	0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0	(hex 0800)

I/O check:

Bits	0–3	0 1 0 0
Bit	4	Sequence indicator
Bits	5–7	0 0 0
Bits	8–15	Device address if available

Storage parity error (two log entries):

First entry: 16-bit address (contents of SAR)

Second entry:

Bits	0–3	1 1 1 0
Bit	4	0
Bit	5	Supervisor state
Bits	6–7	Level
Bits	8–11	LAAK (Note 1)
Bits	12–15	CAAK (Note 2)

Program Check

Program check contains two log entries:

First entry: 16-bit address (contents of CIAR)

Second entry:

Bits	0–3	1 1 0 0 (processor ISA check) 1 1 0 1 (specification check) 1 1 1 1 (storage protect check)
Bit	4	0
Bit	5	Supervisor state
Bits	6–7	Level
Bits	8–11	LAAK (Note 1)
Bits	12–15	CAAK (Note 2)

Notes:

1. LAAK (last active address key) contains the last address key (ISK, OP1K, or OP2K) used.

2. CAAK (current active address key) contains the current instruction space key (ISK) used.

Priority Interrupt Entries

Bits	0–3	0 1 1 0 (exception check)
		0 1 1 1 (attention and exception check)
Bit	4	0
Bit	5	Supervisor state
Bits	6–7	Level
Bits	8–15	Device address

Operate I/O Entries

Bits	0–3	0 0 1 0 (busy after reset check)
		0 0 1 1 (command reject check)
		0 1 0 1 (interface data check)
Bit	4	0
Bit	5	Supervisor state
Bits	6–7	Level
Bits	8–15	Device address

Index

IBM Series/1 4956 Processor Models E and E10
Description

Order No. GA34-0289-1

READER'S COMMENT FORM

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you. Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate.

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page.)

Note: Staples can cause problems with automated mail sorting equipment.
Please use pressure sensitive or other gummed tape to seal this form.

GA34-0289-1
Printed in U.S.A.

Reader's Comment Form

Cut or Fold Along Line

Fold and tape　　Please Do Not Staple　　Fold and tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS　　PERMIT NO. 40　　ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

International Business Machines Corporation
Information Development, Department 28E
1803 (Internal Zip)
P.O. Box 1328
Boca Raton, Florida 33429-9960

Fold and tape　　Please Do Not Staple　　Fold and tape

IBM®

IBM®

International Business Machines Corporation

GA34-0289-1

GA34-0289-1

Printed in U.S.A.